Towards a smart fall detection system using wearable sensors

I Putu Edy Suardiyana Putra

A thesis submitted as part of a Cotutelle programme in partial fulfilment of
Coventry University's and Macquarie University's
requirements for the Degree of
Doctor of Philosophy

March 2018

Coventry University

Faculty of Engineering and Computing

Macquarie University

School of Engineering

Statement of candidate

I certify that the work in this thesis entitled "Towards a smart fall detection system using wearable sensors" has not previously been submitted for a degree nor has it been submitted as part of the requirements for a degree to any other university or institution other than Coventry University and Macquarie University.

I certify that the thesis is an original piece of research and it has been written by me.

I certify that all information sources and literature used are indicated in the thesis.

In addition, I declare that my research has full University Ethical approval and evidence of this has been included within my thesis/submission.

I Putu Edy Suardiyana Putra

To my wife, son, daughter, brother, and parents, who always support me no matter what.

Abstract

A fall-detection system is employed in order to monitor an older person or infirm patient and alert their carer when a fall occurs. Some studies use wearable-sensor technologies to detect falls, as those technologies are getting smaller and cheaper. To date, wearable-sensor-based fall-detection approaches are categorised into threshold- and machine-learning-based approaches. A high number of false alarms and a high computational cost are issues that are faced by the threshold- and machine-learning-based approaches, respectively. The goal of this thesis is to address those issues by developing a novel low-computational-cost machine-learning-based approach for fall detection using accelerometer sensors.

Toward this goal, existing fall-detection approaches (both threshold- and machine-learning-based) are explored and evaluated using publicly accessible datasets: Cogent, SisFall, and FARSEEING. Four machine-learning algorithms are implemented in this study: Classification and Regression Tree (CART), k-Nearest Neighbour (k-NN), Logistic Regression (LR), and Support Vector Machine (SVM). The experimental results show that using the correct size and type for the sliding window to segment the data stream can give the machine-learning-based approach a better detection rate than the threshold-based approach, though the difference between the threshold- and machine-learning-based approaches is not significant in some cases.

To further improve the performance of the machine-learning-based approaches, fall stages (pre-impact, impact, and post-impact) are used as a basis for the feature-extraction process. A novel approach called an event-triggered machine-learning approach for fall detection (EvenT-ML) is proposed, which can correctly align fall stages into a data segment and extract features based on those stages. Correctly aligning the stages to a data segment is difficult because of multiple high peaks, where a high peak usually indicates the impact stage, often occurring during the pre-impact stage. EvenT-ML significantly improves the detection rate and reduces the computational cost of existing machine-learning-based approaches, with an up to 97.6% F-score and a reduction in computational cost by a factor of up to 80 during feature extraction. Also, this technique can significantly outperform the threshold-based approach in all cases.

Finally, to reduce the computational cost of EvenT-ML even further, the number of features needs to be reduced through a feature-selection process. A novel genetic-algorithm-based feature-selection technique (GA-Fade) is proposed, which uses multiple criteria to select features. GA-Fade considers the detection rate, the computational cost, and the number of sensors used as the selection criteria. GA-Fade is able to reduce the number of features by 60% on average, while achieving

an F-score of up to 97.7%. The selected features also can give a significantly lower total computational cost than features that are selected by two single-criterion-based feature-selection techniques: SelectKBest and Recursive Feature Elimination.

In summary, the techniques presented in this thesis significantly increase the detection rate of the machine-learning-based approach, so that a more reliable fall-detection system can be achieved. Furthermore, as an additional advantage, these techniques can significantly reduce the computational cost of the machine-learning approach. This advantage indicates that the proposed machine-learning-based approach is more applicable to a small wearable device with limited resources (e.g., computing power and battery capacity) than the existing machine-learning-based approaches.

Acknowledgements

First and foremost, praise and thanks to Ida Sang Hyang Widhi Wasa, the Almighty, for his showers of blessings throughout my research work, so that I can successfully complete my study.

I would like to express my deep and sincere gratitude to my supervisors, Dr. James Brusey, Associate Professor Rein Vesilo, and Professor Elena Gaura for the support, feedback, and encouragement they have given me.

As my principal supervisors, James and Rein have guided and helped me focus on the research questions to keep me on the right track. Feedback from James and Rein on my research has been invaluable. Furthermore, James has helped me to improve my writing and presentation skills.

I would especially like to thank Elena for giving me the opportunity to work with such great researchers at the Cogent Labs.

I also would like to thank Coventry University and Macquarie University for giving me an opportunity to get into a co-tutelle program, which allowed me to do my research in two different universities, two different countries, and even two different continents! What an amazing experience!

I would like to thank my wife, my son, my daughter, my parents, and my brother for their love, kindness, and support. I wouldn't be where I am today without their backing over the years.

Finally, I would like to thank Dr Keith Imrie for helping me to proofread my thesis chapters, and my close friends who gave me support when times got tough.

Contents

List of Figures

List of Tables

List of Algorithms

Abbreviations

ADL	Activity of daily living
ANOVA	Analysis of variance
CART	Classification and regression tree
CC	Computational cost
CCA	Cascade classifier approach
CPU	Central processing unit
DT	Decision tree
EMA	Exponential moving average
EvenT-ML	Event-triggered machine-learning approach for fall detection
FNSW	Fixed-size non-overlapping sliding window
FN	False negative
FNR	False-negative ratio
FOSW	Fixed-size overlapping sliding window
FP	False positive
FPR	False-positive ratio
GA	Genetic algorithm
GA-Fade	Genetic-algorithm based feature selection for fall detection
GB	Gigabyte
***k*-NN**	k-nearest neighbour
LOSOCV	Leave-one-subject-out cross-validation
LR	Logistic regression
LTS	Long-term support
MEMS	Micro-electromechanical system
PC	Personal computer
RBF	Radial basis function
RFE	Recursive feature elimination
RMS	Root mean square
ROC	Receiver operating characteristic
SMA	Signal magnitude area
SVM	Support vector machine
TN	True negative
TP	True positive
WHO	World Health Organization

Chapter 1

Introduction

Falls can cause several types of injury, including fractures, open wounds, bruises, sprains, joint dislocations, brain injuries, or strained muscles [138]. These fall-related injuries are critical, especially for older people, as they get weaker because of ageing. The World Health Organization (WHO) [121] reports that falls are the second-leading cause of injury-related deaths worldwide. In the UK, falls are the main cause of disability and death for people aged over 75 years [3], while in Australia the number of fall-related hospitalisation patients was relatively high at around 96,000 people in the financial year 2011–2012 [81].

Nowadays, a fall-detection system is employed to notify nurses or healthcare emergency services when a patient has fallen. Based on Igual *et al.* [79], a fall-detection system is defined as: "*an assistive device whose main objective is to alert when a fall event has occurred*". Although a fall-detection system cannot prevent falls, it may alleviate or reduce complications by going some way toward ensuring that fall victims receive help quickly. Also, fall detection and alerting systems must work autonomously, as the fall victim may be unable to trigger an alarm [56]. Being left unattended after a fall can be a serious problem for older people. An older person who has experienced an unattended fall is more likely to be hospitalised with diagnoses of volume depletion, gastrointestinal bleeding, urinary-tract

infections, pneumonia, decubitus ulcer, myocardial infarction, and chest pain [145]. Moreover, based on Tinetti *et al.*'s study [145], non-institutionalised fall victims who were unable to get up were reported to have a decrease in their ability to do basic activities of daily living for three consecutive days after experiencing falls. This means that help must be provided as soon as the victim experiences a fall to reduce complications.

Current studies in fall detection have shown that using wearable sensor technology can give promising results [10, 24, 25, 34, 37, 43, 44, 47, 50, 64, 80, 92, 120, 127]. Advances in wearable sensor technologies, as an impact of the development of micro-electromechanical systems (MEMS), mean that wearable sensors are getting smaller and need less power. Although the development of the hardware is improving rapidly, that does not mean that the fall-detection system is getting better.

Based on Igual *et al.* [79], accelerometer-based fall-detection systems are categorised into two types: threshold-based approaches and machine-learning based approaches. The threshold-based approaches utilise manually-defined thresholds, where some studies use a pattern search approach to optimise their threshold [153, 152], to distinguish between falls and activities of daily living (ADLs), while the machine-learning-based approaches implement machine-learning techniques (e.g., decision tree (DT), k-nearest neighbour (k-NN), and support vector machine (SVM)) to build a classifier. This study has identified some knowledge gaps in existing fall-detection studies using wearable sensors, which are discussed in the next section.

The rest of this chapter is organised as follows: Section 1.1 discusses knowledge gaps in fall-detection studies. Section 1.2 provides research questions of this study. Sections 1.3 and 1.4 show contributions to knowledge and the methodology of this study, respectively. Section 1.5 shows some publications generated from this study. The thesis structure is explained in Section 1.6.

1.1 The knowledge gaps

Although there are existing fall-detection approaches that can give promising results from the studies mentioned above, the following issues have not been considered in those studies:

- Some existing machine-learning-based approaches use a fixed-size non-overlapping sliding window (FNSW) or a fixed-size overlapping sliding window (FOSW) to segment the accelerometer signal before doing the feature extraction [23, 43, 50, 75, 84, 118, 148]. In fact, this sliding-window technique is widely used by human-activity recognition studies using wearable sensors [20, 74, 94, 122, 107, 113, 114, 131, 142, 151, 159, 160, 161, 162, 164, 169]. This segmentation process is critical, as it can increase the classifier's detection rate [13, 122]. The existing studies [50, 44, 43, 118, 148] empirically choose their window type and size. However, these studies do not provide an analysis to support their choices. Therefore, the impact of the window type and size on the classifier detection rate (false-alarm and undetected-fall rates) remains unclear. This impact analysis is critical, as it can give a guidance to fall-detection system developers to develop their own system, in order to make an improvement on the classifier detection rate.

- Fall-detection approaches are categorised into two classes: threshold- and machine-learning-based approaches [79]. Azis *et al.* [8] showed that machine-learning-based approaches are able to outperform threshold-based approaches. However, their study compares threshold-based approaches with only FNSW-based machine-learning approaches. Also, their dataset is not publicly accessible. In fact, most of the current studies in accelerometer-sensor-based fall detection use their own dataset, where those datasets are not publicly available. This causes a validation issue, because the results of those studies are not comparable and the tests of those studies are hard to reproduce [33, 32, 80].

Thus, it is not clear whether both FNSW- and FOSW-based machine-learning approaches achieve a better detection rate than the threshold-based approach on publicly accessible datasets.

- A fall event consists of several stages: pre-impact, impact, and post-impact [89, 118]. These stages are widely used as a basis to extract features by threshold-based approaches [24, 34, 47, 86, 88, 136, 152]. However, for the sliding-window-based machine-learning approaches [43, 50, 75, 84, 148], using these stages as a basis for feature extraction is not yet explored. In fact, Ojetola [118] and Putra *et al.* [127] showed that extracting features based on fall stages (pre-impact, impact, and post-impact) for the machine-learning-based approach can give a relatively good detection rate (93% of F-score for Ojetola study and a 93.5% of F-score for Putra *et al.* study), as every stage has its own characteristics. The main problem with extracting features from those stages is that it is hard to estimate the beginning and the end of each stage when a sliding window is used, and this issue is not investigated in Ojetola and Putra *et al.* studies. Abbate *et al.* [2] utilised high acceleration peaks to estimate the impact stage. However, peaks also occur during the pre-impact stage as a result of protective actions [82] and during the post-impact stage due to a waist bouncing [2]. These peaks make the data-stream segmentation process even harder, as it can misalign the segment with the fall stages. Another problem that is encountered by the machine-learning-based approaches is a computational-cost issue. Kau *et al.* [92] showed that extracting complex features for the classification process can increase the system's computational cost. Having a high-computational-cost system on a wearable device is a disadvantage because this device has limited resources (e.g., computing power and battery capacity). These computational-cost and multi-peak issues remain unexplored in the previous studies.

- Another way to reduce the computational cost of the system is by reducing the number of features using a feature-selection technique [102, 133]. Based on Guyon *et al.* [72], feature-selection techniques are categorised into three classes: filter-, wrapper-based, and embedded. A disadvantage of the current feature-selection techniques is that most of them are not designed to handle multiple selection criteria (e.g, classification accuracy, feature measurement cost, etc.) [165]. This makes most of these techniques only focus on selecting features that can improve the accuracy, without considering the computational cost. There are high-computational-cost features which can give a high accuracy. As an example, using the tilt angle of the body can give a better detection rate than using a minimum acceleration vector magnitude [118], where the tilt angle of the body is calculated by combining tilt angles from accelerometer and gyroscope using a Kalman Filter. In fact, a Kalman Filter implementation is not suitable for wearable devices with a limited processing unit and memory [143]. A study from Saeedi *et al.* [130] proposed a filter-based fall-detection approach to select features based on the detection rate and energy consumption. However, their study focuses only on non-fall activities. Wang *et al.*'s study [154] proposes a feature-selection technique that is based on the detection rate and energy consumption for fall detection using wearable sensors. They propose a wrapper-based feature-selection technique to reduce the number of features from ten to four features. The problem with this study is that it does not provides a clear explanation regarding the way a feature removed from the subset in each iteration. Also, it does not select the best subset from all evaluated subsets. Another problem with Wang *et al*'s study is that it does not compare the proposed approach with other types of feature-selection techniques. Thus, from their study, it is not clear whether using a wrapper-based technique is effective in finding a subset of features that can give an equal or higher detection rate and lower computational cost than using

filter-based or embedded techniques. The last criterion for feature selection that needs to be considered is the number of sensors. Gjoreski *et al.* [64] show that adding more sensors can increase the system detection rate. However, selecting features from different sensor placements has not been explored in the existing fall-detection studies.

Based on the gaps above, the next section discusses four research questions which drive the work in this thesis.

1.2 Research questions

The main aim of this thesis is to reduce the knowledge gaps in fall-detection studies above. Thus, four research questions have been formulated:

1. **RQ1**- What is the impact of the sliding-window type and size on the classifier detection rate (precision, recall, and F-score) when the machine-learning-based approach is used?

   As the role of the sliding window is critical, because it can affect the quality of the extracted features, this question explores the impact of different sliding-window types and sizes on the classifier's detection rate (precision, recall, and F-score). This question also aims to investigate the advantages and disadvantages of the existing machine-learning-based approaches.

2. **RQ2**- Does the sliding-window machine-learning based approach provide a significantly better detection rate than the threshold-based approach on publicly accessible datasets?

   This question aims to investigate the performance of existing fall-detection approaches (both threshold- and machine-learning-based approaches). Publicly accessible datasets are used to achieve a fair comparison between techniques [80].

3. **RQ3**- Does correctly aligning a segment with the fall stages (pre-impact, impact, and post-impact) and using the state of the body of the subject (active or inactive) to trigger the feature-extraction and classification processes improve both the system's detection rate and reduce its computational cost?

   Based on previous studies [118, 127], extracting features from fall stages can improve the classifier's detection rate. However, estimating the beginning and the end of each stage of a fall remains problematic. Also, using a traditional sliding window can increase the computational cost of the system [127]. Therefore, this question evaluates the use of a state machine to correctly align a segment with fall stages and detect the state of the body to trigger the feature-extraction and classification processes, with aims to both significantly increase the classifier's detection rate and reduce the system's computational cost.

4. **RQ4**- Does a meta-heuristic search technique (genetic algorithm) select features that have a higher detection rate and a lower computational cost than features that are selected by single-criterion-based feature-selection techniques (filter-based and embedded techniques)?

   This question asks whether a meta-heuristic search technique (for example a genetic algorithm) can be used to find a subset of features from different sensor placements, that can give the best trade-off between detection rate and computational cost (multi-criteria-based feature-selection technique). Having an accurate and low-computational-cost fall-detection system is important so that the system can be implemented on a small wearable device with limited resources (for example memory and battery power).

1.3 Contributions to knowledge

This thesis provides three main contributions to knowledge:

1. **A study** of both threshold- and machine-learning-based fall-detection approaches on publicly accessible datasets. This contribution aims to answer RQ1 and RQ2. For the impact of the window size on the classifier detection rate (RQ**1**), increasing the window size of the fixed-size non-overlapping sliding window (FNSW) does not necessarily increase the detection rate (in terms of precision, recall, and F-score), unless the lengths of all falls are fixed and uniform. On the other hand, increasing the overlap of the fixed-size overlapping sliding window (FOSW) can reduce the precision in most cases regardless of the machine-learning algorithm. This study shows that increasing the overlap of the FOSW causes an increase in the number of data overlaps between fall and non-fall activities, where these data overlaps cause a reduction in precision. To answer RQ2, a comparison, using publicly accessible datasets, between a threshold-based approach and sliding-window-based machine-learning approaches is provided. The threshold-based approach can achieve an F-score of up to 88.6%, while the machine-learning-based approaches can achieve an F-score of up to 96.5%. These results show that machine-learning-based approaches achieve a better detection rate than the threshold-based approach. However, the difference is not significant in some cases. This means that the machine-learning-based approach still needs to be improved, so that it can achieve a significantly better detection rate than the threshold-based approach in all cases. More detailed results regarding this investigation are provided in Chapter 4.

2. **An event-triggered machine-learning approach for fall detection (EvenT-ML)**. This contribution provides an answer to RQ3 by proposing a state machine that correctly aligns a segment to fall stages, extracts features based on those stages, and uses the state of the user's body to trigger the feature extraction and classification process. This approach is able to achieve an F-score of up to 97.6%, where this result is significantly better than threshold-

based, FNSW-, and FOSW-based machine-learning approaches in all cases. Extracting features based on fall stages is not yet widely used in the sliding-window-based machine-learning approaches in the literature. However, the results from this study show that extracting features based on fall stages can significantly improve the classifier's detection rate. As an additional advantage, EvenT-ML has a significantly lower computational cost than the sliding-window-based machine-learning approaches. Also, EvenT-ML can solve the multi-peak issue, which makes EvenT-ML achieve a significantly better F-score than an existing fall-stage-based machine-learning approach from Putra *et al.* [127]. Chapter 5 provides an overview of EvenT-ML, together with its performance analysis.

3. **A genetic-algorithm-based feature selection for fall detection using wearable sensors (GA-Fade)**. Extracting features from fall stages (pre-impact, impact, and post-impact) can cause the computational cost of the system to increase, because the number of features increases three times. Feature selection is needed to reduce the dimensionality of the feature space, so that the computational cost of the system can be reduced. Also, discarding unnecessary features can increase the classifier detection rate [167]. GA-Fade is proposed to select features that can give a similar F-score to features that are selected by other feature selection techniques, with a lower computational cost, and this is an answer to RQ4. A comparative study between a wrapper-based (GA-Fade), a filter-based (SelectKBest [125]), and an embedded techniques are implemented for this study. The results show that these three feature-selection techniques are able to select features with comparable results in terms of F-score. However, GA-Fade is superior compared to the other two techniques because it can select a subset of features that can give a significantly lower computational cost. This result also confirms that a wrapper-based feature-selection technique is more effective to select features with multiple criteria

than filter-based and embedded techniques.

Table 1.1 shows a summary of the relationships between the problems, the research questions, and the contributions of this study.

1.4 Research methodology

Much of the work on this thesis is based on experimental work and simulation on existing publicly accessible fall datasets. Figure 1.1 shows the steps in this research. Publicly accessible datasets: Cogent [119], SisFall [140], and FARSEEING [1] were used in this study. Detailed information about the datasets is provided in Chapter 3. All experiments were carried out offline on a personal computer (PC) using well-known software libraries (for example the Scikit-learn machine-learning library using the Python programming language [125]), so that all results provided in this thesis are reproducible (the relevance of this method to real-world sensor motes is further discussed in Section 5.2 on page 111). Comparison studies were also conducted, to evaluate the improvement of the proposed techniques over existing techniques.

To measure the performance of the classifier, metrics other than accuracy are chosen. This is because the number of fall data is less than for other activities (data imbalance), and using accuracy can overvalue the always-negative classifier (a classifier that always classifies all samples to the negative class) [55]. Thus, precision, recall, and F-score are used in this study. A leave-one-subject-out cross-validation (LOSOCV) technique is implemented across this thesis to evaluate the performance of the classifier [62, 120], since it intuitively seems to produce an unbiased estimator of performance with unseen *subjects*. For fall detection, the main source of variation is due to characteristics of the subjects or how sensors are attached rather than, say, the time of day or the temperature in the room. However, it is clear that this estimator may be subject to variance and, to some extent, this can be observed in the variance of the test results obtained from cross validation (i.e., with K-folds or

Table 1.1: The problems (Section 1.1), the research questions (Section 1.2), and the contributions (Section 1.3) of this study.

Problem	Research question	Contribution
The impact of window type and size on a classifier's detection rate (in terms of false alarms and mis-detected fall rates)	**RQ1**	Contribution (1)
An evaluation of threshold and machine-learning based approaches	**RQ2**	Contribution (1)
The computational cost of the sliding-window based machine-learning approach	**RQ3**	**EvenT-ML**
Multi-peak issue in fall-stages based segmentation technique	**RQ3**	**EvenT-ML**
Multi-criteria based feature-selection technique	**RQ4**	**GA-Fade**

K subjects, K different test results are obtained).

The variance in the LOSOCV estimate may endanger conclusions about whether one algorithm outperforms another and this issue has been studied, in the context of K-fold CV by Bengio and Grandvalet [16]. They make it clear that one cannot obtain an unbiased estimate of the variance from ordinary K-fold CV. They do not make a claim about LOSOCV – and it may have better characteristics that K-fold CV. However, until such a result is produced, claims of algorithm superiority based on confidence intervals derived from the variance in LOSOCV results should be cautiously analysed.

Chapter 5 uses a hold-out validation technique as an additional check to measure the effectiveness of using data from young and healthy subjects to detect falls in older people. The hold-out validation technique splits the data into a training and a testing set. In this case, the hold-out validation technique uses the data from young and healthy subjects as the training set and the data from older people as the testing set. A Wilcoxon signed-rank test is applied in order to measure the significance of the improvement of the proposed techniques, because the distribution of precision, recall, and F-score values in this study are not normal (the normality of the variables' distributions are measured using the Shapiro-Wilk normality test). Thus, using the Wilcoxon signed-rank test for this study is appropriate since it does not assume normal distribution or homogeneity of variance, therefore it is safer than the parametric test (e.g., t-test) [42]. Another reason for using the Wilcoxon signed-rank test in this study is that the results that are produced by the cross-validation technique are not-independent [16]. Based on Krauth [97], the Wilcoxon signed-rank test can be used as an alternative for dependent samples.

The simulation software was built by adopting the extreme-programming software development methodology [21, 38]. The extreme-programming development methodology allows short development cycles, where this methodology can increase the productivity of the software. An acceptance test (in this case a unit test) was

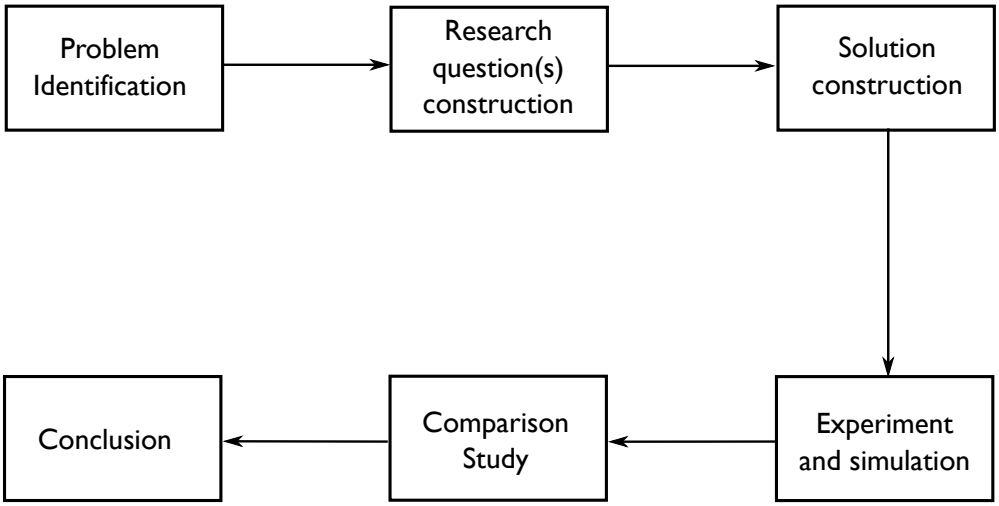


Figure 1.1: Research methodology

implemented iteratively to ensure that all functionalities of the system are tested adequately. All software was built using the Python programming language. Figure 1.2 shows the software development methodology used in this study.

1.5 Publications

This research has fully or partly contributed to the following publications.

1.5.1 Conference papers, poster, and journal articles

- Putra, P. I., Brusey, J., Gaura, E., and Vesilo, R. 2018. An Event-Triggered Machine Learning Approach for Accelerometer-Based Fall Detection. Sensors. https://doi.org/10.3390/s18010020

- Putra, P. I., Brusey, J., and Gaura, E. 2015. A Cascade-Classifier Approach for Fall Detection. In Proceedings of the 5th EAI International Conference on Wireless Mobile Communication and Healthcare (MOBIHEALTH'15), Akram Alomainy, William Whittow, Yang Hao, Konstantina S. Nikita, and Clive G. Parini (Eds.). ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium, pp.

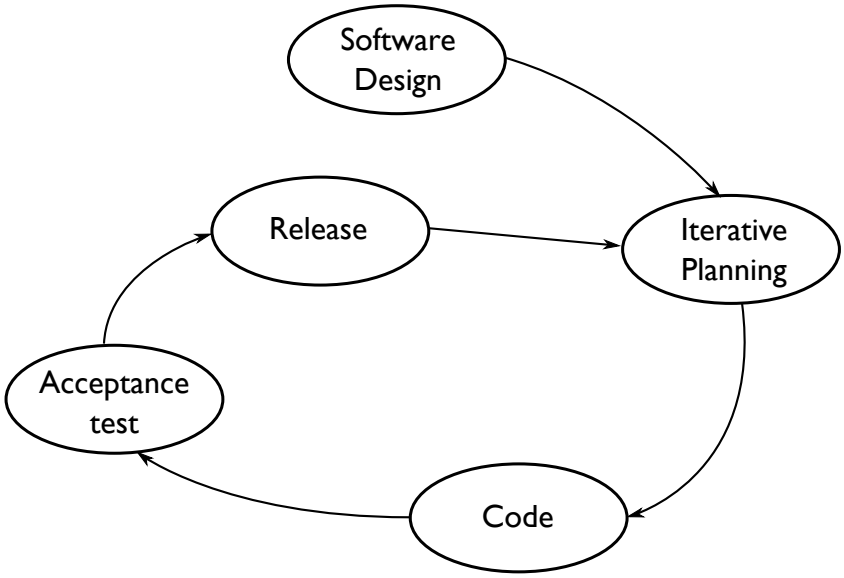


Figure 1.2: Software development methodology. This schematic is a modified version of an extreme-programming methodology from Choudhari and Suman [38]

94–99.

- Putra, I. P. E. S, Brusey, J., Gaura, E. 2015. An intelligent system for fall detection using wearable sensors: issues and challenges, In Proceedings of the 10th International Student Conference on Advanced Science and Technology (ICAST) 2015, Surabaya, Indonesia, pp. 93–94.

- Putra, I. P. E. S, and Vesilo, R. "Genetic algorithm-based feature selection technique for fall detection using multi-placement wearable sensors", In Proceedings of the 12th International Conference on Body Area Networks 2017.

- Putra, I. P. E. S, and Vesilo, R. "Window-size impact on detection rate of wearable sensor-based fall detection using supervised machine learning", In Proceedings of the 1st IEEE Life Sciences Conference 2017.

- Putra, I. P. E. S. A cascade-classifier approach for fall detection. presented as a research poster in the Coventry University Research Symposium, 2014.

1.5.2 Presentations

- The inaugural Macquarie University Research Minds Showcase 2016.
- Sydney Research Bazaar (ResBaz), University of Technology Sydney, 2017 (https://2017.resbaz.com/sydney).

1.5.3 Awards

- Macquarie University Postgraduate Research Funding (PGRF) 2017.

Appendix B details further output resulting from this work, including full copies of the poster, conference and journal papers.

1.6 Thesis structure

Figure 1.3 shows a breakdown of contribution chapters of this thesis. This chapter provides the background, methodology, research questions, and information about the contributions to knowledge of this thesis. Several existing studies are discussed in Chapter 2. More detailed information regarding the Cogent, SisFall, and FARSEEING datasets are provided in Chapter 3. Chapter 4 investigates the performance (in terms of precision, recall, and F-score) of both the threshold-based and machine-learning-based fall-detection techniques. A comparison study between threshold- and machine-learning-based studies is also provided in this chapter. The event-triggered machine-learning-based approach (EvenT-ML) is described in Chapter 5 together with a comparison study between EvenT-ML, traditional, and state-of-the-art machine-learning-based fall-detection techniques. Chapter 6 investigates the use of a genetic algorithm (GA-Fade) to select sub-features from several sensor placements. Chapter 7 gives answers to the research questions, provides conclusions, and proposes prospective future work.

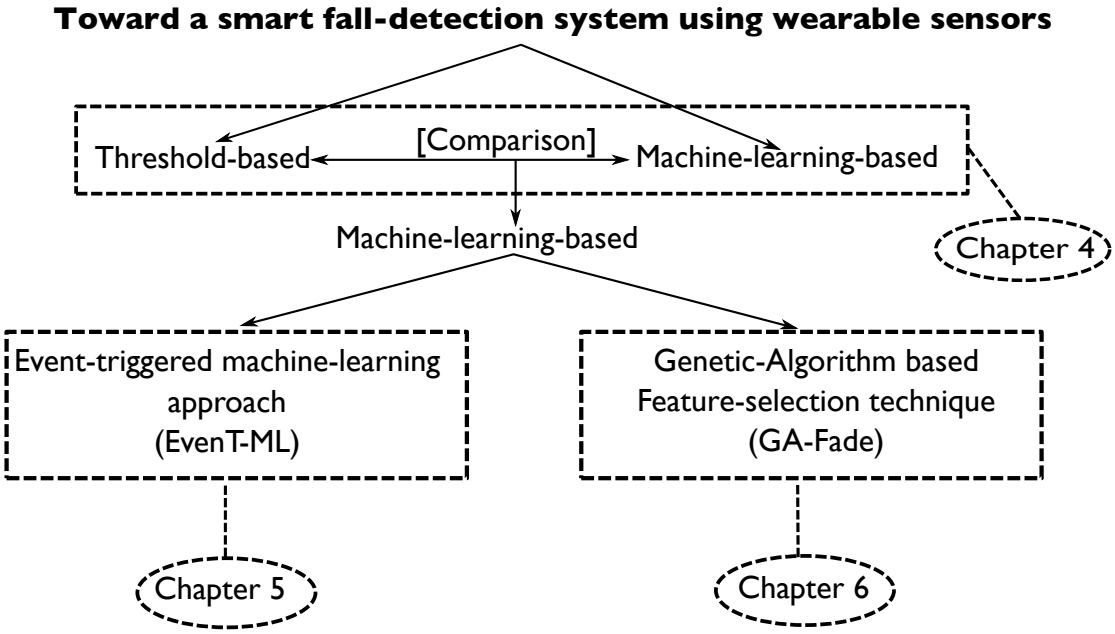


Figure 1.3: Thesis schematic

Chapter 2

Literature review

This chapter discusses existing studies related to falls in older people, fall-detection systems, wearable-sensor-based fall-detection systems, and issues in developing intelligent fall detectors using wearable sensors.

2.1 Falls in older people

2.1.1 Definition of falls

According to the World Health Organization (WHO) [121], a fall is an event that results in a person abruptly coming to rest on the floor or other lower level. Moylan and Bender [111] define a fall as an unexpected positional change, that causes the patient to come to rest on the ground, floor, or other lower surface. Tuunainen *et al.* [146] define falls as coming to the ground or a lower level abruptly, with or without loss of consciousness. Liu and Cheng [104] define a fall as an action when the centre of gravity of the body descends quickly. Based on these several definitions, it can be concluded that falls happen unexpectedly. They can cause the centre of gravity of the body to descend quickly, which makes the subject come to rest on the ground or other lower level with or without maintaining consciousness.

Falls can cause victims to suffer from several physical consequences, such as: fracture, open wound, bruise or blood extravasation, sprain, joint dislocation, brain injury, and muscle strain [138]. Some complications such as hypothermia and pneumonia can be a long-term negative effect of falls [45].

2.1.2 Stages in falls

Paoli *et al.* [124] show that falls have 4 main stages: initial free-fall, impact, motionless, and position change. The initial free-fall state is a weightlessness phenomenon, where the vector sum of the acceleration decreases below $1g$. The impact state is a moment when the body hits the ground, and the motionless state is an inactive condition following the impact phase. The position change is a condition where the body changes posture after a fall. Kangas *et al.* [86] define falls into 4 phases that are: beginning of the fall, falling velocity, fall impact, and posture after fall. However, later in their study, Kangas *et al.* [89] determine that falls have three stages: start of the fall, impact, and horizontal end posture. Similarly, a study from Ojetola [118] summarises a fall into 3 stages:

- Pre-impact: In this stage, the subject experiences a loss of balance. Though, a high velocity is not detected during this stage in real falls when the subject falls from a standing posture [85].
- Impact: The moment when the subject hits the floor or an object. A high acceleration signal characterises this stage.
- Post-impact: This is the stage when the subject is inactive after making contact with the floor, the ground, or an object. The length of this stage varies, depending on the subject's ability to get up after falling.

Becker *et al.* [15] propose 5 phases of fall that are pre-fall, falling, impact, resting, and recovery. The differences between Becker *et al.*'s phases and Ojetola's phases are the pre-fall and recovery phases. The pre-fall phase indicates any activities (e.g.

walking, climbing stairs, or running), while the recovery phase is usually indicated by a centre-of-mass (COM) movement after the victim has been resting (inactive). This recovery phase is used to reduce the number of false alarms because the subject may not need any emergency help after experiencing a fall. Since the datasets used in this thesis do not include the recovery phase in their protocol, this phase is not considered. Thus, this thesis uses the fall stages proposed by Ojetola's study.

Because of the inability to get up is common in the post-impact stage [56], the presence of an automated fall-detection system can be useful for older people. A study conducted by Brownsell and Hawley [27] on the user acceptance of automatic fall detectors shows that 19 out of 21 subjects who wore an automated fall detector were happy that they have fall detectors in their home. Moreover, that study also shows that 18 subjects consider that their safety is increased because of the presence of the automated fall detectors. Although the number of subjects who prefer having a fall-detection system in their home is relatively small, Brownsell and Hawley's study indicates that the presence of automated fall detectors increases older people's confidence. The next sub-section discusses several existing fall-detection approaches.

2.2 Fall-detection approaches

According to Igual *et al.* [79], a fall-detection system is an assistive device whose main objective is to issue an alert when falls occur. The increased number of fall-detection approaches makes it important to have a classification of them. Igual *et al.'s* study categorises fall-detection systems into two classes: context-aware systems and wearable-sensor-based systems. The idea of context-aware systems is to deploy some sensors/devices (for example: cameras [4, 78, 40, 112, 132], floor sensors [5, 53, 110, 129, 147], radar [46, 51, 60, 134], Kinect [12, 14, 36, 61, 101, 103, 106, 139, 128, 166], infrared ceiling sensor [144], or thermal array and ultrasonic sensor [6]) into the user's environment, and using them to detect falls. Some problems of using these

systems are that they are not portable (they cannot detect falls that happen outside the sensor-equipped environment), they might violate the user's privacy (especially for the camera-based systems), and the sensors' price is high (for example a high-resolution camera). On the other hand, wearable-sensor-based systems use mostly accelerometers and gyroscopes to detect human movements and activities. These systems have several advantages, which are that they are more portable, cheaper, and less intrusive (not implanted).

Igual *et al.*'s [79] study shows that the accelerometer-sensor-based fall-detection approaches have two categories: threshold- and machine-learning-based approaches. The threshold-based approaches use pre-defined thresholds to distinguish falls and common activities of daily living (for example: walking, sitting, running, etc.) [24, 86, 136, 47]. The thresholds are generated based on recorded acceleration data from both falls and activities of daily living (ADLs). The second category is machine-learning-based approaches, which use machine-learning algorithms such as Decision Tree [120], Support Vector Machine (SVM) [156, 43, 91, 92], Logistic Regression (LR) [118], or k-Nearest Neighbour (k-NN) [76] to train a classifier to classify falls. On the other hand, based on Pannurat *et al.* [123], the fall- detection approaches are categorised into three: threshold-, machine-learning-, and rule-based approaches. The rule-based approach is basically a multi-threshold-based technique which involves several thresholds in a particular order, while the machine-learning-based approach is based on or partly based on machine learning. This means that the machine-learning algorithm can be used for building the core classifier or optimising the manually-defined thresholds from a hand-designed threshold-based decision-tree classifier (for example: [91, 92]).

This thesis categorises fall-detection approaches into three: threshold-, machine-learning-, and threshold-machine-learning approaches. Since the number of fall detection approaches that use only a single parameter is limited [140, 141], the rule-based approaches are merged with threshold-based approaches in this study.

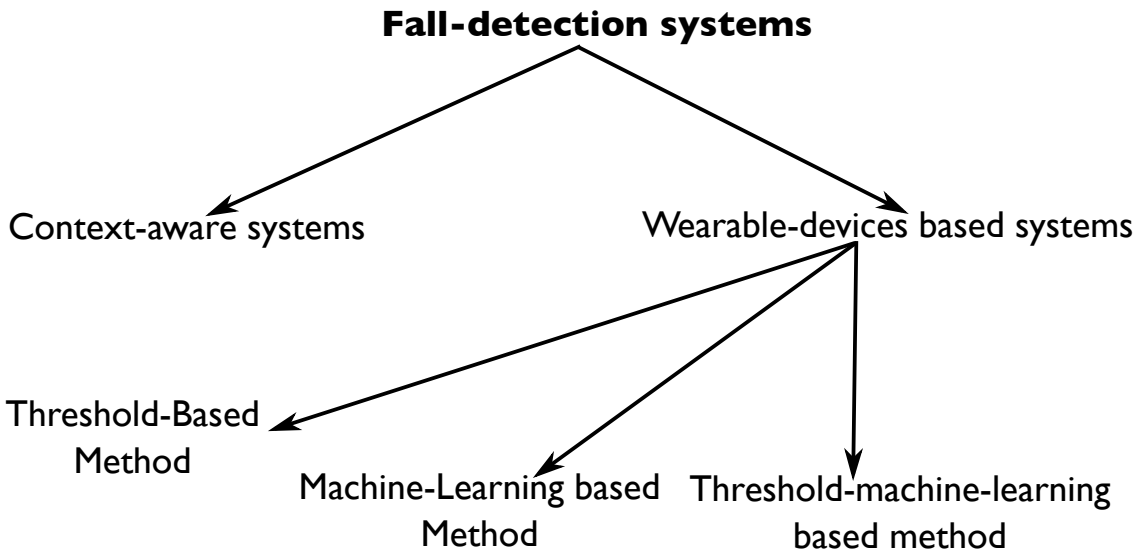


Figure 2.1: Fall-detection systems taxonomy

Most of the threshold-based approaches manually pre-define their threshold, while some studies use heuristic-search approaches to optimise their threshold [153, 152]. This category is a combination of threshold-based approaches and machine-learning-based approaches. Examples of this category can be found in Gjoreksi *et al.* [64], Kau *et al.* [91], and Putra *et al.* [127]. Gjoreski *et al.*'s threshold-based part detects high acceleration, which is one of the characteristics of fall events, while the machine-learning-based part is used to build a classifier that can detect human posture to indicate a fall. A study from Kau *et al.* tries to improve the efficiency of machine-learning-based approaches by using a cascade-style classification using a state machine. Putra *et al.* developed another cascade-style classifier, which uses the state of the body (active or inactive) to trigger the feature-extraction and classification processes. This means that the feature extraction is done when the subject's body is moving about in an active way.

In summary, a taxonomy of fall-detection systems is shown in Figure 2.1. The next section reviews several current approaches in fall detection using wearable sensors.

2.3 Wearable-sensors-based fall detection

This section discusses current approaches in fall detection using wearable-sensor technology. This section has three parts: threshold-based, machine-learning-based, and threshold-machine-learning-based approaches.

2.3.1 Threshold-based fall-detection approaches

Table 2.1 gives a summary of fall-detection studies that use threshold-based approaches. It is written in ascending order in terms of published year. Threshold-based approaches use pre-defined thresholds to detect falls, where these thresholds are usually generated based on recorded data.

Chen *et al.* [34] in 2005 conducted a study on fall detection using a custom-made accelerometer sensor. Their study shows that the minimum acceleration of fall events is $3g$ (g is the acceleration due to gravity). However, they also found that several activities such as running, jumping, and sitting abruptly might produce an acceleration that is similar to that of fall events. As an additional finding, their study identified that fall events consist of three different stages: free-fall, impact, and dampening effect. The first stage (free-fall) involves a small dip in the acceleration. The impact stage is the stage that has a high peak of acceleration, while the dampening-effect stage is the stage when the subject lands and remains lying on the floor. These fall stages are important for future study in fall detection, as they show that falls have a particular pattern that is not possessed by most other activities. Though, the performance of the fall-detection algorithm is not explicitly reported, another problem that arises from this study is that the thresholds were determined by empirically analysing the dataset. The results are biased since the thresholds are determined using all samples, which causes the classifier to "see" the optimum value to detect fall events beforehand.

A method for defining thresholds was proposed by Kangas *et al.* [87] in 2007.

Table 2.1: Summary of papers on threshold-based fall-detection approaches using wearable sensors

Authors	**Hardware Platform (Sampling frequency)**	**Sensor Placement**	**Number of Subjects & Ages (years)**	**Fall Types**
Chen *et al.*, (2005) [34]	ADXL210E 2D accelerometer	waist	2 (Ages: N/A)	N/A
Kangas *et al.* (2007) [87]	3 triaxial accelerometers	waist, wrist, head.	2 (Ages: 22 and 38)	forward, backward, lateral.
Kangas *et al.*, (2008) [86]	3D accelerometer (400 Hz)	waist, wrist, head	3 (Ages: 38–48)	forward, backward, lateral.
Bourke *et al.*, (2010) [25]	3D accelerometer (200 Hz)	waist	10 (Ages: 24–35); 10 (Ages: 73–90).	forward, backward, lateral.
Sorvala *et al.*, (2012) [136]	ADXL345 3D accelerometer, LPR530 & LY530AL 3D gyroscopes. (50 Hz)	waist, ankle	2 (Age: 26)	forward, backward, lateral, fail to sit.
Dumitrache and Paşca, (2013) [47]	LIS3LV02DQ 3D accelerometer (40 Hz)	waist	1 (Age: 25)	N/A
Kangas *et al.* (2015)[88]	3D triaxial accelerometer (ADXL330) (50 Hz)	waist	16 (Ages: 88.4 ± 5.2)	N/A
Wang *et al.* (2016) [152]	digital accelerometer (ADXL362) and barometric pressure sensor LPS25H.	chest (in a shape of pendant worn on a lanyard around the neck)	First part: 11 (Ages: 25.9±1.7) Second part: 5 (Ages: 26.2 ±1.3)	forward, backward, lateral.
Sucerquia *et al.* (2017) [140]	triaxial accelerometer (ADXL345)	Waist	Young subjects: 23 (Ages: 19-30) Older subjects: 15 (Ages: 60-75)	forward, backward, lateral.

This method utilises box-plots to determine the threshold. This study shows that its proposed technique is able to achieve 100% of both sensitivity and specificity. The sensitivity (recall) and specificity are calculated by

$$\text{Sensitivity} = \frac{\text{True positive (TP)}}{\text{TP + False Negative (FN)}}, \tag{2.1}$$

$$\text{Specificity} = \frac{\text{True negative (TN)}}{\text{TN+False positive (FP)}},$$

where sensitivity and specificity are used to measure false-negative and false-positive ratios, respectively. Similar to Chen *et al.*'s study, this study uses all samples to determine its thresholds to detect falls. This makes the results of this study is extremely biased. Also, from this study, it can be seen that data overlaps between falls and other activities exist. These data overlaps exist because some non-fall activities share similar characteristics with falls and they can cause a high number of false alarms or false negatives (mis-detected falls).

Kangas *et al.* [86] in 2008 conducted another study in fall detection using three threshold-based approaches, where they compare those approaches using simulated activities. Three types of falls: forward, backward, and lateral falls, were recorded from three healthy adult subjects. The approaches in Kangas *et al.*'s study were developed based on stages in fall events: beginning of the fall, falling velocity, fall impact, and posture after fall. This study uses several parameters: sum vector of acceleration from three axes, sum vector of high-pass filtered data, vertical acceleration, difference between the maximum and minimum acceleration in a 0.1 s sliding window, and velocity. Based on their experiments, the approach that uses 3 stages (beginning of the fall, falling impact, and posture after fall) achieves the highest accuracy. This study adapts thresholds from Kangas *et al.*'s [87] study, which makes the results of this study less biased than studies from Kangas *et al.* [87] and Chen *et al.* [34]. However, this study only measures the sensitivity of the system and ignores the specificity of the system. This makes the performance of the system in reducing

the number of false alarms unclear. In fact, having a high number of false alarms can cause a rejection by users [115].

In line with Kangas *et al.*'s study, Bourke *et al.* [25] conducted a comparison study of existing threshold-based approaches in 2010. In their study, Bourke *et al.* used data from healthy, young and older subjects. This study involves a total of 10 older people with ages 73–90 years. However, the older subjects were only asked to do several scripted and unscripted (real) ADLs, and real-fall data from the older people were not available for the experiment. The Bourke *et al.* study uses the following parameters as thresholds: maximum acceleration, velocity, and posture. The best parameter combination to detect simulated falls is the maximum acceleration and velocity, with a 100% sensitivity and a 98.9% specificity. According to the experiment on the unscripted ADLs, Bourke *et al.*'s experiment shows that the number of false alarms ranged from 0.945–45.34 false-alarms/day. This shows that the threshold-based approach produces a relatively high number of false alarms.

Sorvala *et al.* (2012) [136] proposed two threshold-based approaches that reduced the number of false alarms. Their experiment uses two healthy subjects with sensors strapped to their ankle and waist. A lesson learned from this study is that extracting information (body posture) during the impact stage can reduce the number of false alarms. The results show that their proposed approach achieves up to 95.6% of sensitivity and 99.6% of specificity. However, they do not provide specific information regarding the protocol used in their study. This makes the results of this study hard to reproduce. Similarly to Kangas *et al.* [87] and Chen *et al* [34], Sorvala *et al.*'s study also determines the thresholds using samples from all subjects, which makes the results biased. Also, this study shows that data overlaps between falls and non-falls exist.

In 2013, Dumitrache and Paşca [47] proposed a threshold-based fall-detection approach that uses six thresholds: peak value, base length (the length of a fall activity), ratio between peak and base length (R_1), post-impact velocity (V), ratio

between V and R_1 (R_2), and post-impact activity level. They claim that their approach can achieve 97.05% of sensitivity and 99% of specificity as long as all parameters are used to classify the activity. Although their approach is able to achieve relatively good results, the way they determine the thresholds for all the parameters is not explained.

The issue about using all samples in the threshold determination process was discussed by Wang *et al.* [152] in 2016. Thus, in their study, Wang *et al.* split the dataset into a training and a testing set, where the thresholds are determined using the training set. The classifier was built in a binary-decision-tree style, where the thresholds are optimised using a particle-swarm optimisation (PSO) algorithm. The approach proposed in this study is able to achieve 93% of sensitivity and 87.3% of specificity. These results show that the false-alarm and the false-negative rates are not balanced, while keeping the balance between these two rates (low false alarms and low false negatives) is important [115]. Also, this study conducted free-living trials, where the subjects were requested to wear the device while doing their normal activities (without researcher supervision) for a week. The study claimed that its approach can produce fewer false alarms than studies from Bagala *et al.* [10] and Kangas *et al.* [88]. However, comparing the results of fall-detection studies, where these results are taken directly from their paper, is inappropriate because each study involves different types of sensors, subjects, and types of activities. This issue is discussed in Igual *et al.*'s [80] study. To get a fair performance comparison, fall-detection approaches should be implemented and compared on the same dataset. Thus, the availability of a publicly accessible dataset for studies in this field of research is important. Another issue that is discussed in Wang *et al.*'s study is the power consumption of the device. This study claimed that their approach can make the device run for 664.9 days on average. Lessons learned from this study: (1) the thresholds should be determined from a training set, so that the classifier is not exposed to the testing set; and (2) the energy consumption is an important aspect

to be considered in designing a fall-detection approach.

Sucerquia *et al.* (2017) [140] tried to solve the publicly accessible dataset availability issue by making their dataset publicly accessible. Sucerquia *et al.* used ten-fold cross-validation to evaluate their proposed fall-detection approach. Some features are used to distinguish falls from other activities, and this study uses a sliding window to extract features. However, the exact size of the window remains unclear. The only information provided is that the optimum window lies between 200 ms and 2 s. In fact, Chapter 4 shows that the size of the window is critical since it can affect the classifier's performance.

A study that involved real-fall data from older patients was conducted by Kangas *et al.* [88] in 2015. In their study, Kangas *et al.* implemented their custom-made device and proposed a fall-detection approach for older patients who live in older-people care units. To detect falls, this study implemented approaches from their previous study [86, 89]. This study shows that their fall-detection approach can achieve 80% of sensitivity with a false-alarm rate as low as 0.025 per usage hour. However, the data from this study are not publicly accessible.

This subsection shows several existing threshold-based fall-detection approaches together with their advantages and disadvantages, The Sub-section 2.3.2 provides some information regarding some existing machine-learning-based approaches.

2.3.2 Machine-learning-based approaches

Table 2.2 gives a summary of fall-detection studies that use threshold-based approaches. It is written in ascending order in terms of published year. The machine-learning-based fall-detection approaches use a machine-learning algorithm (for example: Naive Bayes, Decision Tree (DT), Logistic Regression (LR), Support Vector Machine (SVM), or k-Nearest Neighbour (k-NN)) to build a classifier to distinguish falls from ADLs.

Choi *et al.* [37] in 2011 conducted a study on fall detection using a machine-

Table 2.2: Summary of papers on machine-learning-based and threshold-machine-learning-based fall-detection approaches using wearable sensors

Authors	**Hardware Platform (Sampling frequency)**	**Sensor Placement**	**Number of Subjects & Ages (years)**	**Algorithms**	**Fall Types**
Choi *et al.*, (2011) [37]	3D accelerometer, 2D gyroscope	chest	N/A	Naive Bayes (NB)	forward, backward, lateral
Gjoreski *et al.*, (2011) [64]	3D accelerometer (6 Hz)	chest, waist, right thigh, right ankle	11 (Ages: N/A)	threshold-based + unspecified data-mining algorithm	falling fast to the ground, falling slowly, falling from chair
Ojetola *et al.*, (2011) [120]	Shimmer sensors: 3D accelerometer & 3D gyroscope (100 Hz)	chest, thigh	8 (Ages: 21–33)	C4.5 Decision Tree	forward, backward, lateral
Abbate *et al.*, (2012) [2]	Shimmer sensors: 3D accelerometer & 3D gyroscope (100 Hz)	waist	3 (Ages: N/A)	threshold-based + neural network	forward, backward, and faint
Erdogan and Bilgin, (2012) [50]	MTS310CB Board	waist	N/A	k-Nearest Neighbour (k-NN)	falling down without specific direction
Diep *et al.*, (2013) [43]	ADXL330 accelerometer (100 Hz)	waist	12 (Ages: N/A)	Support Vector Machine (SVM)	N/A
Vallejo *et al.*, (2013) [148].	ADXL345 3D accelerometer	waist	21 (Ages: 18–56)	Artificial Neural Network (ANN)	N/A
Wang *et al.*, (2013) [156].	N/A	right, left up waist; right, front, left waist; right, left-thigh; right, left-knee; right, left-shank; right, left-ankle.	5 (Ages: 21–26)	SVM	forward, backward, lateral
Kau and Chen, (2015) [92].	3D accelerometer on Sony Xperia U-series (150 Hz)	pocket	5 (Ages: N/A)	threshold-based + SVM	fall down
Kambhampati *et al.*, (2015) [84]	triaxial accelerometer on Android phone	waist	6 (Ages: N/A)	Decision Tree NB/ Multi-layer/ Perceptron (MLP)/ SVM	fall front, fall back, fall right, and fall left
Putra *et al.*, (2015) [127].	Shimmer sensors: 3D accelerometer & 3D gyroscope (100 Hz)	chest	48 (Ages: N/A)	threshold-based + J48 Decision Tree/ Logistic Regression (LR) / MLP	forward, backward, lateral
Yuan *et al.*, (2015) [168]	triaxial accelerometer (ADXL345)	wrist	Young subjects: 10 (Ages: N/A) Older subjects: 3 (Ages: N/A)	threshold-based+ C4.5	N/A
Bourke *et al.*, (2016) [168]	McRoberts Hybrid, McRoberts MiniMod, activPAL3, Samsung Galaxy SII, Samsung Galaxy S3, and uSense.	Fifth Lumbar Spine (L5)	Older subject (Ages: N/A)	C4.5	N/A
Hsieh *et al.*, (2016) [77]	triaxial accelerometer (ADXL325)	N/A	Young subjects: 10 (Ages: N/A)	SVM k-NN	N/A
He *et al.*, (2017) [75]	triaxial accelerometer (custom-made device)	back	20 (Ages: 20-45)	Bayes Network	Sideward and backward

learning-based approach. Their study shows that the Naive Bayes classifier can detect falls with a 99.4% accuracy using a sensor node strapped to the chest; the accuracy is calculated using

$$\text{Accuracy} = \frac{(\text{TP}+\text{TN})}{\text{TP}+\text{FP}+\text{TN}+\text{FN}}.$$

This study uses five features that are extracted from the sensor node: accelerations on x, y, z axes, gyroscope on x and y axes. This study also measured the classifier's accuracy when an additional sensor is added, strapped on the subject's thigh. Four additional features were calculated from the thigh sensor: acceleration on x, y, z axes, and from the x-axis of the gyroscope. The classifier can achieve an accuracy of 99.8% by using two sensors (chest and waist) based on this study. However, it is not clear how the sensors were placed on the subject's body since the authors do not provide this information. This becomes an issue because this makes the results of this study are hard to reproduce, and this issue is the weakness of Choi *et al.*'s study. This study also implemented some feature-selection techniques to reduce the number of features used. Although the number of features can be reduced (from 9 features to 3 features), the accuracy also reduces.

A decision-tree-based fall-detection algorithm was proposed by Ojetola *et al.* [120] in 2011. By strapping sensors on chest and thigh, their algorithm achieves an 81.82% precision, a 92.19% recall (sensitivity), and a 99.45% accuracy. The precision is calculated using the following formula

$$\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}}, \tag{2.2}$$

where the precision is another measurement for the false-alarm rate. Following this study, Ojetola [118] conducted another study in 2013, which proposed the use of fall stages (pre-impact, impact, and post-impact) as a basis for extracting features. This study shows that using fall stages as a basis for feature extraction can give better

accuracy. Thus, this thesis adopts the concept of stage-based feature extraction from Ojetola's study. This study also implements the concept of a sliding window to extract features, using a 12 s overlapping sliding window. However, this study does not provide a strong justification to support its window size choice. Another problem with the approach from this study is that it uses an N-1 window overlap size (N = number of samples in a window), which means that the window slides sample by sample. This causes a high computational cost [127]. Also, using a higher window overlap can increase the data overlaps, and can cause an increase in false alarms or false negatives. This data-overlap issue is discussed in Section 4.5. An advantage of this study is that its dataset is publicly accessible. Thus, the dataset from Ojetola's study is used in this thesis. More-detailed information regarding this dataset is provided in Chapter 3.

Erdogan and Bilgin [50] proposed a k-NN based algorithm to detect falls in 2012. To measure the similarity between the segment in the input stream and the segment on the training set, their k-NN-based algorithm uses the Euclidean distance. The experiment shows that this algorithm can achieve 89.4% accuracy, 100% recall, and 85% precision. This study uses a sliding window (with a 7-sample size) to segment the data stream, and features are extracted from each segment. Instances from the segments are used to train and test the k-NN-based classifier. However, the features extracted from each segment are not explained.

Studies from Diep *et al.* (2013) [43] and Wang *et al.* (2013) [156] proposed SVM-based algorithms for fall detection. Diep *et al.* [43] used accelerometers that were embedded on a Wii remote as the detector. For feature extraction, they implemented a sliding window with a size of 1.8 s and an overlap of 0.6 s. These numbers were obtained from their previous study [126], which study focuses on classifying daily activities. In their study, Diep *et al.* classify activities into two classes (binary classification): fall and non-fall, and use an SVM to train their classifier. Their approach is able to achieve up to 91.9% precision and 94.4% recall in a 10-fold

cross-validation evaluation setting, and up to 91.8% precision and 90.34% recall in a leave-one-subject-out evaluation setting. The main problem with this study is that they use the window size from their previous study, which does not involve any fall activities.

Wang *et al.* [156] in 2013 developed an approach that can detect falls with 100% sensitivity and 94% specificity, by using 13 sensors strapped to a subject's body. This approach uses an SVM to build the classifier, with 5 features: maximum resultant acceleration, maximum acceleration of z-axis, minimum acceleration of z-axis, the angle between legs and thighs, and the angle between thigh and torso. Although this approach can achieve 100% of sensitivity, wearing 13 sensors can cause the subject to feel uncomfortable.

Vallejo *et al.* [148] proposed a neural-network-based fall-detection approach in 2013. This approach uses a feed-forward network to train the classifier using the velocities on the x, y, and z axes as features. These features were extracted using a window of size 10 samples, though there is not a further explanation regarding the window choice. Vallejo *et al.*'s approach is able to achieve a 98.4% sensitivity and a 98.6% specificity under a hold-out evaluation scheme. Also, they conducted a test using unsupervised activity samples recorded during 12 hours, from a subject. The test results show that not a single false alarm is generated.

Kambhampati *et al.* [84] used cumulant-based features to detect falls together with their direction (forward, backward, or lateral) in 2015. This study also uses a sliding window with a size of 0.25 s to extract its features. Similar to Ojetola's study [118], this study uses an N-1 window overlap, which can cause a computational-cost issue. This study claims that their approach is able to receive an accuracy up to 97.32% by using an SVM-based classifier.

Bourke *et al.* [23] proposed a decision-tree-based (C4.5) classifier fall-detection approach in 2016, where this approach was evaluated using real-fall data from older subjects. This study uses a triaxial accelerometer to detect impact and a gyroscope

to detect the angle posture of the subject. This study indicates that detecting the impact stage of a fall is critical because some features are extracted based on this stage. For example, this study uses angular velocity from the gyroscope as a feature, where this feature is extracted 0.5 s before and after the impact peak. This approach can perform well if the signal is pre-segmented and the impact point is manually defined. However, in a real-time situation, it is hard to automatically segment the data and define the impact stage. In fact, Figure 2 from their paper shows two peaks during a fall, which can confuse the classifier. Later in this thesis, this phenomenon is defined as multi-peak issue. For this type of approach, detecting the impact stage precisely is mandatory.

Another sliding-window-based approach was proposed by He *et al.* [75] in 2017. This approach uses a 2 s sliding window to extract features, where these features are used to train and test a Bayes-Network-based classifier. The authors claim that their approach is able to achieve up to 95.67% of accuracy, 99.00% of sensitivity, and 95% of specificity. Having a static window (especially a short one) for recognising an activity, where this activity is executed for a long period of time, cannot give a high accuracy for the classifier. This is because the window produces many identical consecutive temporal windows with similar features, which makes the classifier classify the same instances from a particular activity over and over again [122].

Several studies above show that using machine-learning approaches can give promising results. However, the main issue of the machine-learning-based approaches above is the window size and window type (overlap and non-overlap). Every study has their own window size, where this window size is defined based on their own dataset (not a publicly accessible dataset) or taken from previous study, making the results of this study hard to justify. Also, each study uses different types of features (although some of the features overlap between studies), which means that the impact of the window and overlap sizes to the classifier's performance remains unclear. Important information that is achieved from this subsection is that extract-

ing features based on fall stages (pre-impact, impact, and post-impact) may give a better accuracy, since each stage has important characteristics that can distinguish falls from other activities.

To improve the effectiveness and efficiency of the existing machine-learning-based fall-detection approaches, several studies developed threshold-machine-learning-based approaches [64, 92, 127]. These studies apply some manually defined thresholds before/after classifying the activity using a machine-learning-based classifier, in order to improve the detection rate or reduce the computational cost of the system. Subsection 2.3.3 provides a review of several studies that combine threshold-based algorithms and machine-learning algorithms into a threshold-machine-learning-based algorithm.

2.3.3 Threshold-machine-learning-based approaches

This section provides some reviews of studies that use combinations of the threshold-based and machine-learning-based approaches. The purpose of these approaches is to improve the accuracy or/and reduce the complexity of the system.

Gjoreski *et al.* [64] in 2011 split their approach into two modules. The first module is fall detection and is followed by posture recognition. This module uses a supervised machine-learning algorithm to build the classifier to identify the posture of the subjects. Their study classifies the data samples into 7 postures: standing, sitting, lying, standing up, going down (sitting down, lying down, or falling), on-all-fours position, and sitting on the ground. To recognise the posture of the subject in real time, some features are extracted from a data stream using a 6 s sliding window. The results show that combining a threshold-based approach and a machine-learning-based approach can achieve an accuracy of 94% on average. The study also shows that attaching more sensors on the subject's body can increase the detection rate of the classifier. To detect a fall, this algorithm detects the maximum acceleration during a 1 s window. If the maximum acceleration exceeds a threshold,

then the posture is identified. However, the way to define the threshold value or the value itself for this process is not provided. Another problem about this approach is the multi-peak issue. Sometimes multiple peaks are produced by an accelerometer sensor due to a protective action during the fall [82] or a bouncing effect [2]. Because the posture recognition is started after the peak is detected, these multiple peaks can cause a confusion regarding when to start the posture recognition. In this case, the timing is important.

Abbate *et al.* [2] in 2012 developed a smart-phone-based fall-detection system which combines threshold- and machine-learning-based approaches. In their study, they considered the multi-peak issue as an important matter. Thus, they proposed a mechanism to detect multiple peaks by using a finite state machine. Their study claims that the proposed approach together with a neural-network-based classifier is able to achieve 100% of sensitivity and 100% of specificity. This proposed technique mainly uses the finite state machine just to avoid an unnecessary feature extraction by only extracting features when a fall-like event is detected by the finite state machine. This idea is adopted by Putra *et al.* [127] to develop a cascade-classifier approach for fall detection.

Putra *et al.* [127] proposed another cascade approach in 2015. This approach uses the state of the body to trigger the training/testing process of the machine learning (this idea was inspired by Abbate *et al.*'s approach). Also, in their study, Putra *et al.* tested several machine-learning algorithms. This cascade approach adopts a concept of fall stages from Ojetola [118], and is able to achieve up to 93.5% of precision, 94.2% of recall, and 93.5% of F-score on average. An F-score can be calculated using

$$F\text{-}score = \frac{2.\mathrm{TP}}{2.\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}, \tag{2.3}$$

where $F - score$ is the harmonic mean of precision and recall (sensitivity). This approach is able to achieve a significantly better F-score (in their study, Ojetola's approach achieves an 87.4% F-score, which is 6.1% lower than the cascade approach)

and computational cost than the approach proposed by Ojetola. These results show that the cascade-classifier approach that is proposed by Putra *et al.* is able to receive fewer false alarms and false negatives than Ojetola's technique. Although this cascade-classifier approach is able to improve the detection rate of the classifier and reduce the classifier's computational cost, this approach does not solve the multi-peak issue, which becomes more critical for this approach because it uses fall stages as a basis for feature extraction. Mistakenly choosing a peak as an indicator of an impact stage may cause the segment to misrepresent the fall stages. This multi-peak issue is discussed in more detail in sub-section 2.5.4.

Kau and Chen (2015) [92] proposed another threshold-machine-learning-based approach for fall detection called a cascade approach. In their study, they use some thresholds as part of the pre-processing stage, before the training/classification process, where they describe their system as a finite state machine. They implemented a state machine as a representation of their technique. The system uses the thresholds to prevent itself from extracting high-computational-cost features, and it uses the SVM-based classifier to classify falls. Their experiment shows that their proposed approach can detect falls with 92% of sensitivity and 99.75% of specificity. To show their reduction in the computational cost, they implemented their proposed algorithm on a smartphone and investigated the energy consumption of their system. The results show that their system is able to alleviate the power-consumption burden of the device, though the amount of energy reduction that is given by implementing their proposed state machine is not explicitly mentioned.

Hsieh *et al.* [77] proposed a threshold-machine-learning-based approach in 2016, where this approach uses two thresholds: maximum acceleration vector magnitude and maximum acceleration vector magnitude on the horizontal plane. If the acceleration value exceeds these two thresholds, a feature extraction is done using an FOSW. In this study, the authors tested different sizes of the window: 0.1s–0.5s, with a window overlap of 50%. k-NN and SVM are used to train the classifier. They

claimed that their proposed approach is able to achieve a 96.26% accuracy when k-NN is used to train the classifier. In this study, fall stages (free-fall, impact, and resting on the ground) are used as a basis to extract the features. In the training phase, the identification of the fall stages is done manually. This is possible since the data are already pre-segmented. In the testing phase, they have a fixed template pattern that has to be matched with the accelerometer signal. Once the acceleration signal matches the template pattern, features are extracted based on the fall stages. The main problem of using this template pattern is that the system does not know how to correctly align the template pattern on the continues acceleration data stream. Misaligning the template with the accelerometer signal can cause the system misses a fall event.

Although most of the approaches explained above give a promising result, there are some issues that still exist. Section 2.5 explains four important issues regarding fall-detection approaches using wearable sensors.

2.4 Machine-learning algorithms

This thesis considers four machine-learning algorithms: decision tree (DT), k-Nearest Neighbour (k-NN), Logistic Regression (LR), and Support Vector Machine (SVM). Based on the studies from Ojetola [118], Erdogan and Bilgin [50], and Kau *et al.* [92], these machine-learning algorithms have been shown to provide a relatively high accuracy, though using k-NN can give a higher computational cost during the classification process.

This thesis uses modules from the Scikit-learn library [125], so that the results can be reproduced. Note that the Scikit-learn library provides the decision tree algorithm as a Classification and Regression Tree (CART). CART is similar to the C4.5 algorithm [125], which gives a relatively good detection in Ojetola's study [118]. The following subsections give an overview of the machine-learning algorithms used.

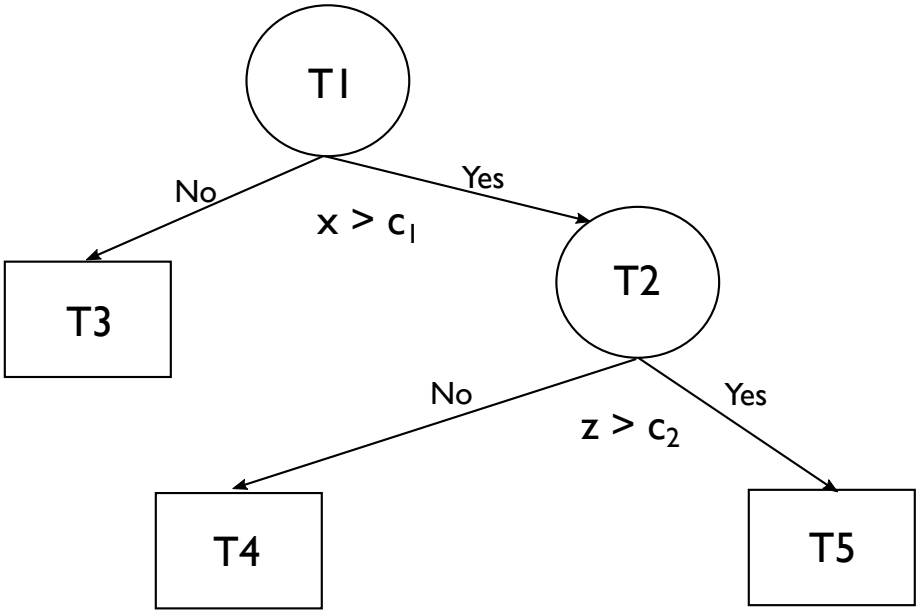


Figure 2.2: An example of a classification and regression tree (CART). x and y are predictors, while c_1 and c_2 are thresholds

2.4.1 Classification and Regression Tree (CART)

CART was first proposed by Breiman *et al.* [26] and has been used for about 33 years. CART builds a tree by recursively partitioning the data into smaller pieces. Figure 2.2 shows an example of CART. T1 is defined as the root of the tree, while T2 is called *a non-terminal node* and T3, T4, T5 are defined as *terminal nodes.* In a classification case, T3, T4, and T5 are associated with classes. x and z take a role as a predictor, and both c_1 and c_2 are thresholds. Then, $(x > c_1)$ or $(z > c_2)$ is called a split. For a classification task, x and z are associated with the features used. Based on Breiman *et al.* [26], three important elements in constructing the tree are:

- How the best split is selected.
- When to decide to stop splitting a *non-terminal node.*
- How to assign a class to terminal nodes.

To build a tree, CART follows these steps [18, 26]:

(1) CART evaluates all possible splits from all predictors (features), then picks the best split among all the splits from all predictor variables. The "best" split can be defined as a split that can most reduce the impurity. An impurity function is a function ϕ defined on $P = \{p_1, p_2, ..., p_J\}$ where J is the total number of classes used. For example, in a binary classification case, the value of J is equal to 2 ($J = 2$). P satisfies: $p_j \geq 0$, where $j = 1, 2, 3, ..., J$ and $\sum_{j=1}^{J} p_j = 1$. The function ϕ has the following properties:

1. ϕ is a maximum when all classes are equally mixed together ($\frac{1}{J}$, $\frac{1}{J}$, ..., $\frac{1}{J}$).
2. ϕ is a minimum when only one class achieves 1 while the others achieve 0: $(1, 0, 0, ..., 0)$, $(0, 1, 0, ..., 0)$,..., $(0, 0, 0, ..., 1)$.
3. ϕ is a symmetric function of $p_1, p_2, ..., p_J$.

Given an impurity function ϕ, to measure the impurity (i) of a node t, use

$$i(t) = \phi\left(p(1|t), p(2|t), ..., p(J|t)\right). \tag{2.4}$$

In this study, ϕ is defined as the Gini index

$$i(t) = \sum_{j=1}^{J} p_j(1 - p_j). \tag{2.5}$$

Then, to measure the impurity reduction (Δi) of a split s on t, use

$$\Delta i(s, t) = i(t) - p_R i(t_R) - p_L i(t_L), \tag{2.6}$$

where:

- t_R is the right child node,
- t_L is the left child node,
- p_R is the proportion of the data samples in t belonging to t_R, and

- p_L is the proportion of the data samples in t belonging to t_L.

(2) The second step is partitioning the data based on the best split.

(3) The second step is repeated until all samples have been placed in terminal nodes.

Beside the Gini index, ϕ also can be defined as the Bayes error, or the cross-entropy function [18]. The Gini index is favoured as it tends to split a node into one small pure node and one large impure node [18, 26].

2.4.2 *k*-Nearest Neighbour (*k*-NN)

Algorithm 2.1 shows the classification process using k-NN. In this study, the Minkowski distance is used. The following formula is used to calculate the Minkowski distance between two points P_1 at (x_1, y_1) and P_2 at (x_2, y_2).

$$Minkowski\ distance = (|x_1 - x_2|^m + |y_1 - y_2|^m)^{1/m}, \tag{2.7}$$

where $m = 2$. For this study, three numbers are used for k: 1, 2, and 3. Note that the Minkowski distance with $m = 2$ is equal to the Euclidean distance.

Algorithm 2.1 k-Nearest Neighbour (k-NN) algorithm

1: For every instance (x) in testing set:
2: Calculate distance between x and all points in training set.
3: Sort the distances in increasing order.
4: Take the k points that have the shortest distance to x.
5: Find the majority (tie-breaks) class/label among those points.
6: Return the majority class as the classification result.

2.4.3 Logistic Regression (LR)

Logistic Regression (LR) is a technique which allows the classifier to estimate categorical outcomes (can be 2 or more categories) from different predictors, where those predictors can be either categorical, continuous, or both [54]. This can be done by using

$$P(Y|x) = \frac{1}{1 + e^{-(\omega_0 + \omega_1 x_1 + ... + \omega_n x_n)}},$$

where $P(Y|x)$ is a function to estimate the probability of class Y given x, x_n is the predictor, and ω_n is a weight (sometimes called a regression coefficient). For the two-class case, if $P(Y|x) > 0.5$, then this outcome is categorised as 1 (or True) by the classifier. On the other hand, if $P(Y|x) \leq 0.5$, then the classifier categorises the outcome as 0 (False). For this study, Y is a binary class that represents Fall (True) and Non-Fall (False), while $X = \{x_1, x_2, ..., x_n\}$ is the set of features used. For the Scikit-learn library, regularisation is implemented to avoid the model/classifier remembering the training data (called data overfitting). To get the value of ω_i for a feature x_i, LR with L2 regularisation minimises the following cost function

$$\underset{\omega,c}{min} \frac{1}{2}\omega^T\omega + C\sum_{i=1}^{n} \log(\exp(-y_i(X_i^T\omega + c)) + 1),$$

where n, c, and C are the number of instances, an intercept of the LR, and an inverse of the regularisation strength (or penalty parameter [52]), respectively. A smaller C indicates a stronger regularisation, where this regularisation is useful to reduce data overfitting. Since the value of C does not change during the training process (C is a hyper-parameter) and the search space for this value is extremely large, some values of C are arbitrarily chosen: $10^8, 10^9$, and 10^{10}. More detailed information about the library can be found in Scikit-learn [125] and Fan *et al.* [52].

Table 2.3: Kernel function used [71, 93, 125]

Kernel function	Type of classifier
$K(x, x') = (x^T x')$	Linear kernel
$K(x, x') = \Big[(x^T x') + 1\Big]^d$	Polynomial kernel
$K(x, x') = \exp\Big(- \parallel x - x' \parallel^{2/\gamma}\Big)$	Gaussian radial basis function (RBF)

2.4.4 Support Vector Machine

A support vector machine (SVM) trains its classifier by maximising the margin between the classes in the training set [22]. Training examples that are close to the decision boundary are called support vectors, and the decision boundary is called a hyperplane. Given n training inputs $X = x_1, x_2, x_3, ..., x_n$ with labels $Y = \{-1, 1\}$,

$$\text{where} \begin{cases} Y_k = 1, \text{if } x_k \in class\, A \\ Y_k = -1, \text{if } x_k \in class\, B. \end{cases} \tag{2.8}$$

then the decision function $D(x)$ is given by

$$D(x) = \sum_{k=1}^{n} \alpha_k K(x_k, x) + b,$$

where $K(x_k, x)$, b, α_k are the pre-defined kernel, the error bias, and coefficients that need to be adjusted during the training process. This thesis uses 3 kernels (Table 2.3): linear, polynomial (with degree 3), and radial basis function (RBF), where d is the degree of polynomial and γ is the inverse of the radius of influence of samples selected by the model as support vectors.

2.5 Issues in developing an automated fall-detection system using wearable sensors

2.5.1 The use of publicly accessible datasets

The sub-sections above show several existing fall-detection approaches. Although most of the studies above show a detection rate of 90% or above, they evaluated their technique on their own datasets, which are not publicly accessible. This is the first problem that needs to be covered by this thesis. Using a publicly accessible dataset is important to get a fair comparison between techniques [80]. Based on Igual *et al.* [80], for example a nearest-neighbour classifier, its performance really depends on the dataset. This type of classifier can only perform well on a particular dataset. Thus, using a non-publicly accessible dataset means that the results of those studies cannot be directly compared.

Chapter 3 of this thesis shows detailed information on the three publicly accessible datasets used in this study: the Cogent [119], SisFall [140], and FARSEEING [1] datasets. Since these datasets provide data from accelerometer and gyroscope sensors, data from other types of sensors (e.g. a barometric sensor [152]) are considered for future work. This thesis focuses on the use of accelerometer sensors because using a gyroscope can increase the cost and the energy consumption of the device [2, 35, 67, 66, 105, 130].

The problem of using datasets that are collected from young subjects is that the results might not represent the performance of the classifier on older patients [10]. However, a study from Jamsa [82] shows that acceleration signals from young subjects share some similarities with acceleration signals from older subjects. Thus, Chapter 5 provides an evaluation of using data from young subjects (the Cogent and SisFall datasets) on the data from older subjects (FARSEEING dataset).

2.5.2 The use of sliding windows in the fall detection/activity-recognition studies using a machine-learning-based approach

Generally, human activity recognition consists of four steps: data preprocessing, segmentation, feature extraction, and classification [155]. For a real-time human-activity recognition system, since wearable sensors (for example an accelerometer) produce a continuous data stream, some activity recognition studies (especially studies that use a machine-learning algorithm to build their classifier) use a fixed-size sliding window to segment the data stream. Then, some features are extracted from each segment, where those features are used to classify the activity. This sliding-window technique is widely used to segment data streams by real-time human-activity recognition systems. Table 2.4 summarises sliding-window-based activity recognition systems using wearable sensors. The segmentation process is critical, as it can produce differentiable feature values, which can increase the classifier's detection rate [19, 69]. In view of the fact that falls are part of human activity, this segmentation process becomes an important matter to be investigated.

Based on Banos *et al.* [13], the sliding window is one of two types: Fixed-size Non-overlapping Sliding Window (FNSW) and Fixed-size Overlapping Sliding Window (FOSW). Figures 2.3 and 2.4 show illustrations of FNSW and FOSW respectively in a real-time situation, where the system does not have any knowledge regarding the start or the end of an activity.

A serious problem arises when the length of the sliding window does not fit the length of the activity. A study from Gu *et al.* [69] shows that an error in segmenting a data stream of an activity can seriously affect the detection accuracy. For example, a short window can truncate an activity while a large window can overlap two consecutive activities. The current studies of fall detection/activity recognition using sliding-window-based machine-learning approaches rely on the figures from previous

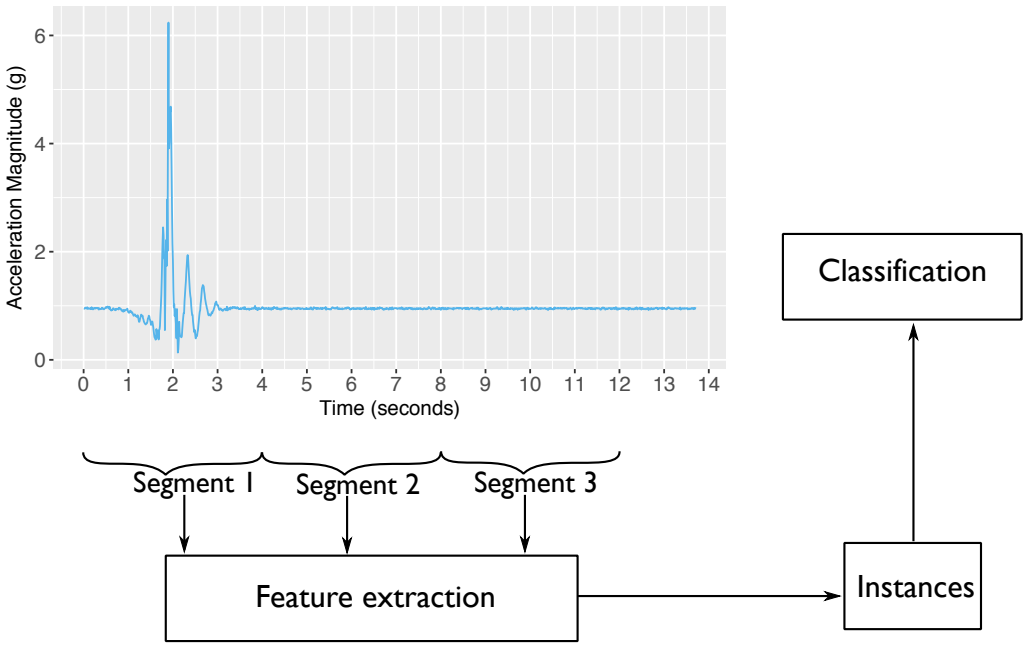


Figure 2.3: Illustration of the use of FNSW in feature extraction

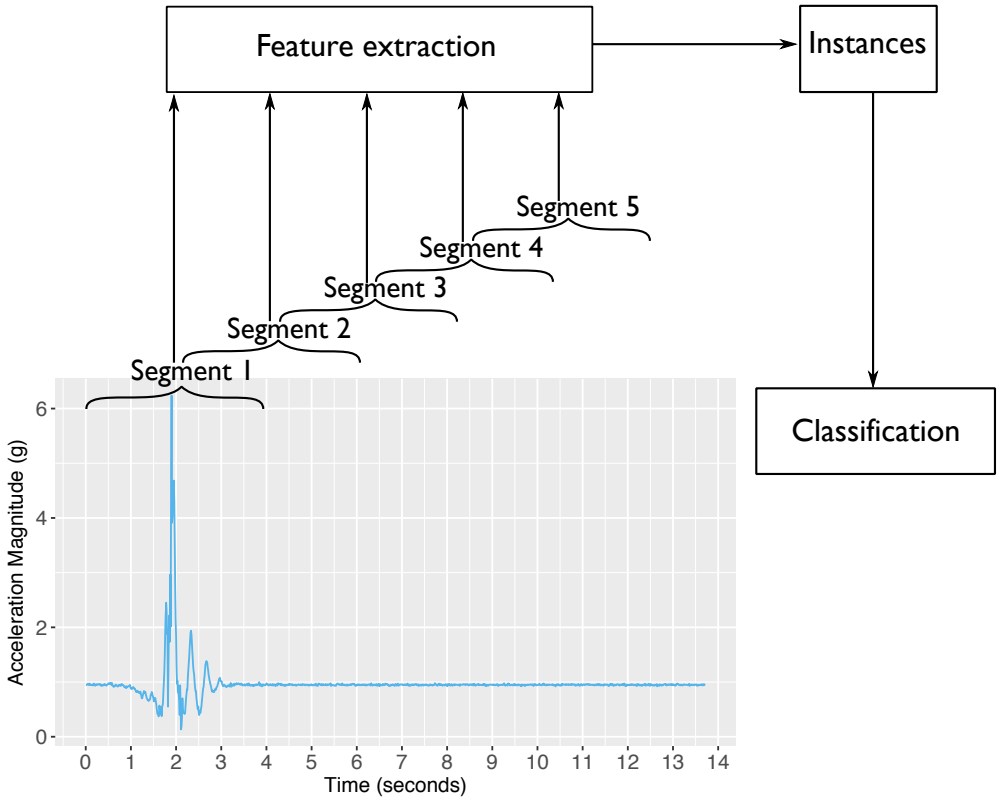


Figure 2.4: Illustration of the use of FOSW (with 50% overlap) in feature extraction

studies [13]. Thus, the impact of the window size on the detection rate, especially for fall detection, is not clear. Although Bersch *et al.* [19] provide a comprehensive study on the impact of the sliding-window length and overlap size for activity recognition, they do not include any fall activity. Therefore, an analysis of the impact of a sliding window on the classifier's performance is needed, and this becomes the second issue. This analysis is also important as it can be a basis to make a performance improvement of the machine-learning-based approach. Chapter 4 covers an investigation of this issue.

2.5.3 Threshold-based approach vs machine-learning-based approach

The third issue in fall-detection studies is an investigation into threshold-based and sliding-window-based machine-learning approaches' performance. Vallejo *et al.* [148] show that generating thresholds for a fall detector is difficult, as there are overlaps between fall and non-fall data. However, their study does not provide any comparative analysis between threshold- and machine-learning-based approaches in terms of the detection rate. A study from Aziz *et al.* [8] shows that machine-learning-based approaches provide a better overall recall and specificity than the threshold-based approach. However, their dataset is not publicly accessible and their study does not include the FOSW-based machine-learning approach. Thus, a performance comparison between threshold- and sliding-window-based machine-learning approaches on a publicly accessible dataset has not been conducted. This issue is addressed in Chapter 4.

Table 2.4: Sliding-window-based activity recognition system using wearable sensors

Authors	Window type (FNSW/FOSW/dynamic)	Window size / Window overlap size	Algorithm(s)
Yang *et al.* (2008) [164]	FOSW	5.12 s / 50%	Multilayer feedforward neural network
Khan *et al.* (2010) [94]	FNSW	3.2 s / -	ANN
Mannini *et al.* (2010) [107]	FOSW	6.7 s / 50%	Hidden Markov Model (HMM)
Laguna *et al.* (2011) [122]	dynamic	- / -	Dynamic Bayesian Network
Bhattacharya *et al.* (2014) [20]	FOSW	1 s / 50%	En-Co-Training [70]
Wan *et al.* (2015) [151]	dynamic	- / -	Naive Bayes, Bayesian Network, C4.5, Naive Bayes Tree, and HMM
Ni *et al.* (2016) [113]	FOSW	2.5 s / 50%	Random Forest, *k*-NN, MLP, NB, J48, and SVM.
Wen and Wang, (2016) [161]	FOSW	2.56 s / 50% and 5 s / 50%	AdaBoost
Xu *et al.* (2016) [162]	FOSW	5.12 s / shifted by 1s	Multi-layer feed-forward neural network
Wannenburg and Malekian (2017) [160]	FOSW	1 s / 50%	*k*-NN and *k*-Star
Noor *et al.* (2017) [114]	dynamic	- / -	Threshold-based
Savvaki *et al.* (2017) [131]	FOSW	128 samples / 50%	*k*-NN and SVM
Sztyler *et al.* (2017) [142]	FOSW	1 s / 50%	Random forest
Wang *et al.* (2017) [159]	FOSW	0.3 s / 50%	Mixed and Reduced kernel extreme learning model (M-RKELM)
Yurtman *et al.* (2017) [169]	FNSW and FOSW	5 s (FNSW) / 2.56 s and 50% overlap (FOSW)	Bayesian Decision Making, *k*-NN, SVM, ANN
Hassan *et al.* (2018) [74]	FOSW	2.56 s / 50%	Deep Belief Network

2.5.4 Data segmentation for a stage-based machine-learning approach for fall detection

Ojetola [118] and Putra *et al.* [127] show that extracting features based on fall stages can improve the detection rate. However, estimating the beginning and the end of each stage in a real-time setup has not been investigated by these studies. Studies from Abbate *et al.* [2], Bai *et al.* [11], Putra *et al.* [127], and Ojetola [118] use high acceleration peaks to determine the beginning of the impact stage (the moment when the body hits an object). Although it is simple in theory, using peaks as an indicator for the impact stage during a real-time implementation is challenging due to the multi-peak issue. In fact, multiple peaks appear in real fall data from an older subject [85].

The multi-peak issue (the fourth issue), which is caused by protective actions when the victim falls [82], can cause a misleading determination of the impact stage. To explain this issue, approaches from Ojetola [118] and Putra *et al.* [127] are chosen as an illustration. Both approaches use a 2 s non-overlapping window to detect the state of the human body (active/inactive). If there are peaks exceeding a threshold (1.6g) during the 2 s window, another 12 s window is placed around the highest peak (P') of that 2 s window and that peak is determined as the impact moment, where this can cause a confusing determination of the impact stage. The illustration of this process is shown in Figure 2.5a. Some readers might argue that this misleading problem can be solved by finding the highest peak during the 12 s window, and determining that highest peak as the impact moment. However, this solution can only be used when a larger window is used. Figure 2.5b shows an illustration of a situation when a shorter window is used; the impact stage is truncated and the post-impact stage cannot be captured. Although using a larger window size can solve the issue for this situation, it can be a disadvantage because the system needs to "wait" longer to get more samples. Note that this multi-peak issue exists only in

a real-time situation, while aligning the window with the impact stage can be easily done in an offline mode.

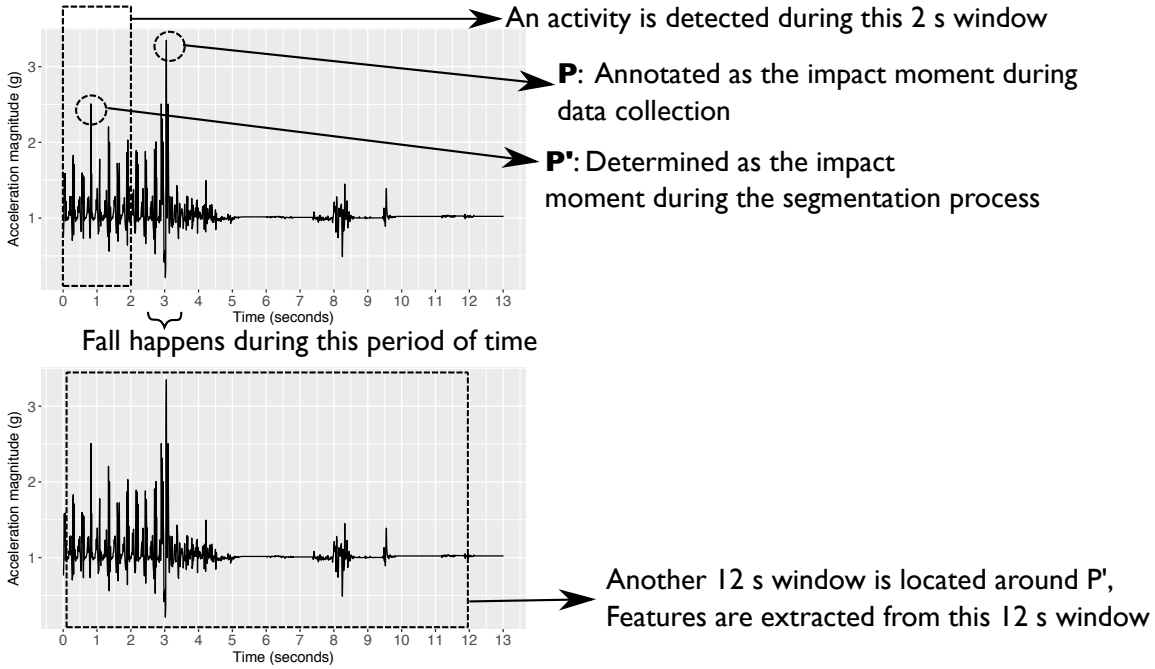
In fact, this multi-peak issue is shown in Abbate *et al.*'s study [2] (this issue is referred to as a bouncing problem), where multiple peaks usually appear from the waist bouncing after a victim falls from bed. In their study, Abbate *et al.* proposed an approach that can solve the multi-peak issue hoping to reduce the number of false alarms, though the accuracy of their approach is quite low with 61.6% of accuracy, 90.9% of sensitivity, and 31% of specificity.

To solve this multi-peak/misalignment issue while improving the detection rate of the classifier, this thesis proposes a novel machine-learning-based approach. This approach is able to solve the multi-peak issue and to correctly estimate the beginning and the end fall stages (pre-impact, impact, and post-impact). Chapter 5 provides more detailed information regarding this novel approach.

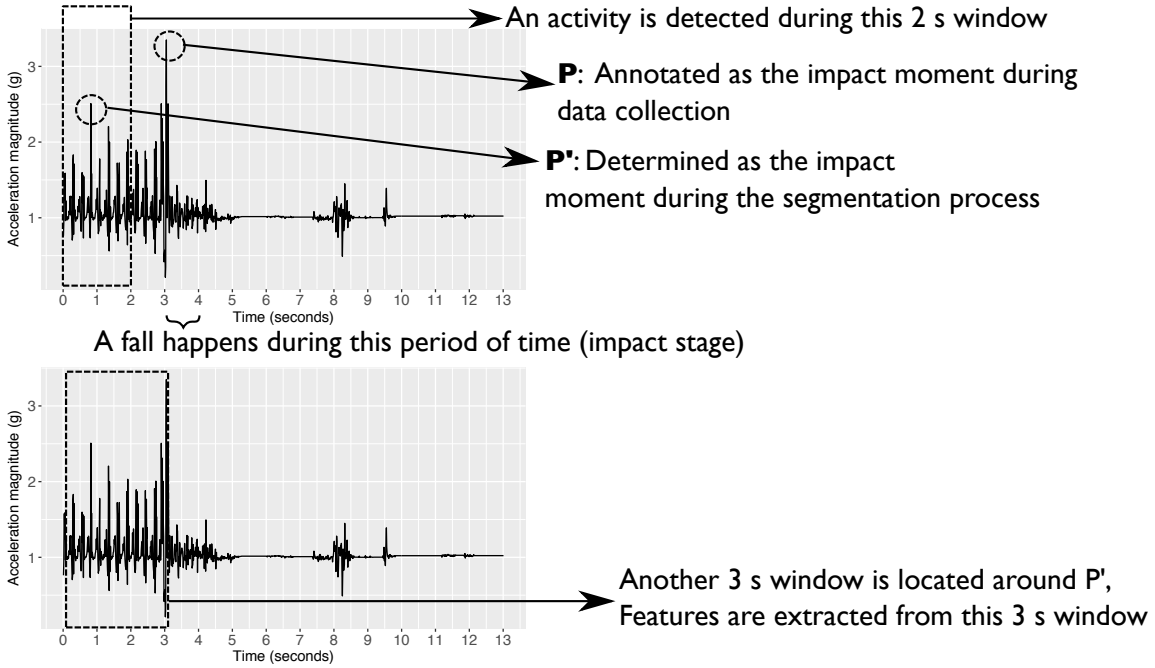
2.5.5 Feature-selection technique for fall detection

In theory, using more features can increase the classifier's detection rate [167]. However, using more features can make the learning process slower or even can reduce the classifier detection rate, as there may be some irrelevant or redundant features [167]. Moreover, extracting more features can also increase the system's computational cost [92], which can increase the energy consumption of the device. Therefore, the number of features needs to be reduced by using a feature-selection technique.

Broadly speaking, feature-selection techniques consist of three categories: wrapper, filter, and embedded. Both the wrapper and embedded techniques involve a learning algorithm for selecting features. The wrapper methods [102, 158] involve a learning algorithm and choose a subset of features based on the machine-learning performance, while the embedded methods select features during the training process of the learning algorithm [73]. The filter methods [7] do not have any dependency on learning algorithms. To select a subset of features, the filter methods require



(a) A segmentation process with a long window [120, 127]



(b) A segmentation process with a short window

Figure 2.5: Multi-peak issue illustrations

less computation than the wrapper methods. However, the filter methods appear to ignore features that can give more information when they are used together [158]. This disadvantage can reduce the accuracy of the classifier. The main drawback of the wrapper methods is that they have a higher computational cost than the filter-based methods [158], as the wrapper methods need to test all possible feature subsets and select a subset of features that can give the optimal accuracy. Doing an exhaustive search in a feature space of N features requires an evaluation of 2^N possible feature combinations [99].

To further reduce the computational cost of the fall-detection system, studies from Li *et al.* [102] in 2015 and Wang *et al.* [158] in 2016 implemented a feature-selection technique to reduce the dimensions of the features. Li *et al.* [102] use a Bayes framework and receiver-operating-characteristic (ROC) curve to select features. Their technique is able to achieve 86.08%, 94.31%, and 95.75% accuracies with 4 features, 8 features, and 12 features, respectively. On the other hand, Wang *et al.* [158] proposed a game-theory-based feature selection with a k-NN and an SVM as the machine-learning algorithms. Their approach can achieve up to 74.42% of accuracy with 21 features. The proposed feature-selection techniques from Li *et al.* [102] and Wang *et al.* [158] only focus on selecting features that can give an optimal detection rate, without considering their computational cost. This is because most of the existing feature-selection techniques are designed to select features based on only a single criterion, namely the detection rate. In fact, there are features that improve the accuracy but have a high computational cost. For example, the Ojetola [118] study shows that the tilt angle of the body, where this is calculated by combining tilt angles from an accelerometer and a gyroscope using a Kalman Filter, gives a better F-score than using a minimum acceleration-vector magnitude. It is obvious that finding a minimum acceleration-vector magnitude requires less computation than calculating tilt angles using a Kalman Filter. Also, implementing a Kalman Filter in wearable devices can be difficult, as these devices have limited

processing units and memory [143]. Furthermore, having a high-computational-cost system can drain the battery of the device quickly [92]. Thus, a fall-detection system needs features that can give a maximum detection rate with a minimum computational cost.

Saeedi *et al.* [130] in 2014 proposed a filter-based feature-selection technique, where this technique can select features from different sensor locations (waist, wrist, arm, and ankle), that can give a relatively good accuracy and low computational cost. Their feature selection technique can select features that can achieve a 70% to 99% accuracy and save an 88% to 99.% of energy. However, this study does not include fall activities.

Wang *et al.* [154] in 2017 proposed a wrapper-based feature-selection technique that considers both the accuracy and the power consumption of the device. They claim that their proposed approach is the first feature-selection technique that considers both accuracy and power consumption. The main idea of their technique is to remove a feature in each iteration when the energy consumption and classification error reduce. However, they do not provide a clear step for selecting a candidate-removed feature for each iteration. The next limitation of their approach is that it does not select the best feature subset from all evaluated feature subsets. This is because it does not provide a mechanism to compare the results of the final output with all results from the previously evaluated feature subsets. Another problem with Wang *et al.*'s study is that they do not provide any comparative analysis between their proposed technique and other feature-selection techniques from other categories (filter based and embedded). There is a chance that using filter-based or embedded techniques can give better or similar results than the wrapper method. Since using either filter-based and embedded techniques requires less time to run than the wrapper method, it is better to use one of these techniques rather than the wrapper method when the results are similar.

Chapter 6 proposes a genetic-algorithm-based feature-selection technique (GA-

Fade), where this technique is able to choose low-computational-cost features from several different sensor placements, where those features can give an optimum detection rate. A comparative analysis between GA-Fade, filter-based (SelectKBest), and embedded (Recursive Feature Elimination) techniques [125] is provided.

2.6 Chapter Summary

This chapter reviews some definitions of falls together with their stages and some existing approaches in fall-detection studies. A taxonomy of fall-detection systems is provided in this chapter. In general, falls can be defined as an unexpected event that can cause the centre of gravity of the body to descend quickly, which makes the subject come to rest on the ground or other lower level with or without consciousness. A fall consists of three stages: pre-impact, impact, and post-impact.

Fall-detection systems consist of two major classes: context-aware and wearable-devices based systems. The wearable-device based systems can be divided into three classes: threshold-based, machine-learning-based, and threshold-machine-learning-based approaches. Some issues that exist in current fall-detection studies are:

- Most of the existing studies use a non-publicly-accessible dataset, which means that the results of those studies cannot be directly compared. Using a publicly-accessible dataset is recommended to evaluate the fall-detection approaches to get a fair comparison result. Also, using a publicly accessible dataset makes the results provided in this thesis are easy to reproduce.

- Because the way data are segmented can affect the detection rate of a machine-learning-based fall-detection system, it is important to choose an appropriate window type (FNSW or FOSW) and size. However, an investigation on the impact of the sliding-window technique on the detection rate, where this investigation is done using publicly accessible datasets, has not been done.

- A comparison between threshold-based approaches and sliding-window-based machine-learning approaches using publicly accessible datasets has not been done. This comparison is used to determine the advantages and disadvantages of both threshold- and sliding-window-based machine-learning approaches, where these advantages and disadvantages can be used as a basis for improving the detection rate of a fall-detection system.

- Although using fall stages as a basis for feature extraction can increase the system detection rate, it is difficult to define the beginning and the end of each stage because of the multi-peak issue. Also, extracting complex features for the sliding-window-based machine-learning approach can increase the computational cost of the system.

- Because existing feature-selection techniques in fall detection are designed to select features based on only one criterion (detection rate), they cannot select low-computational-cost features that can give an optimum detection rate.

The next chapter discusses the publicly accessible datasets used in this study.

Chapter 3

Falls and activities of daily living datasets

The previous chapter discusses existing studies in fall detection, together with their limitations. This chapter reviews publicly accessible datasets that are used in this thesis. As using a publicly-accessible dataset can give a fair comparison between techniques [80], this study uses three publicly accessible datasets: (1) Cogent dataset[1] [119], (2) SisFall dataset[2] [140], and (3) FARSEEING dataset[3] [1]. Four advantages of using the Cogent and SiSFall datasets are:

- Cogent and SisFall have more subjects than three other publicly accessible datasets: DLR [57], tFall [109], and mobiFall [149].
- The Cogent dataset has near-fall activities, where these activities are mostly mis-detected as falls, which can produce a high number of false alarms in the

[1]The dataset can be downloaded at: `http://skuld.cs.umass.edu/traces/mmsys/2015/paper-15/`.

[2]The SisFall dataset can be downloaded at: `http://sistemic.udea.edu.co/en/investigacion/proyectos/english-falls/`.

[3]This dataset is not fully publicly accessible. Instead, this dataset is available by request. Information related to this dataset request can be found at:`http://farseeingresearch.eu/the-farseeing-real-world-fall-repository-a-large-scale-collaborative-database-to-collect-and-share-sensor-signals-from-real-world-falls/` .

real-world case [100]. These activities are important to evaluate the effectiveness of the fall-detection approach on handling false alarms.

- The Cogent dataset has data from three sensor placements. This can allow fall-detection approaches to be tested on different sensor placements. Using more than one sensor placement has been shown to be able to increase the system's detection rate [64].

- The SisFall dataset has more types of falls. This dataset has 15 different types of falls, which is more fall types than the Cogent, DLR, and tFall datasets.

Another dataset used in this study is the FARSEEING dataset. This dataset contains real falls from older patients, and those falls are important to justify the use of data from young and healthy subjects to evaluate fall-detection approaches, which is debatable. Bagala *et al.* [10] show that using data from younger subjects to determine thresholds for detecting falls in older people is not effective. This is because some factors (such as body mass, age, clinical history, and diseases) might affect the value of the thresholds. On the other hand, a study from Jamsa *et al.* [82] shows that real falls present a similar pattern to laboratory-based falls. Their study also confirms that real forward, sideways, and backward falls show the existence of pre-impact and impact stages. Thus, to support Jamsa *et al.*'s finding, this chapter provides a discussion about the comparison between fall acceleration signals from young subjects and from older patients.

3.1 Cogent dataset

3.1.1 Subject profile

As some data from some subjects are corrupt (incomplete data or inappropriate annotation), this thesis uses 46 subjects from this dataset (see Table 3.1), where those subjects include males and females. Sensors were strapped to the chest and

Table 3.1: Subjects' body profiles from Cogent dataset

Profile	**Subjects with chest and thigh sensors.**	**Subjects with chest, waist, and thigh sensors**
Number of males	37	13
Number of females	9	5
Age (years)	23.5 ± 5.5	22 ± 2.8
Height (cm)	172.7 ± 7.7	172.4 ± 10.1
Weight (kg)	69.7 ± 12.8	66.1 ± 13.7

thigh of each subject. Eighteen subjects (including males and females) had a sensor strapped to their chest, waist, and thigh. The thigh placement is considered because some studies use a smartphone placed in the thigh pocket to detect falls [91, 92].

3.1.2 Hardware

The Cogent dataset used Shimmer sensors with a sampling rate of 100 Hz for its data collection. The sensor consists of a 3D accelerometer, a 3D gyroscope, a Bluetooth device, and an MSP430F1611 microcontroller. More details about Shimmer are provided in [31]. The Shimmer sensors transfer the data to a personal computer (PC) using Bluetooth and these data were manually annotated with LabView. Figure 3.1 shows the sensor placement and a Shimmer device.

3.1.3 Protocol

Each of the subjects staged 14 falls (including 6 forward, 4 backward, and 4 lateral falls) and several ADLs for 23 minutes on average. The length of both falls and ADLs for this dataset varies and Figure 4.2 shows the length of falls of the Cogent dataset. More detailed information regarding the length of both falls and ADLs can be found in Ojetola *et. al.* [119]. In total, this dataset has 644 fall and 1,196 ADL

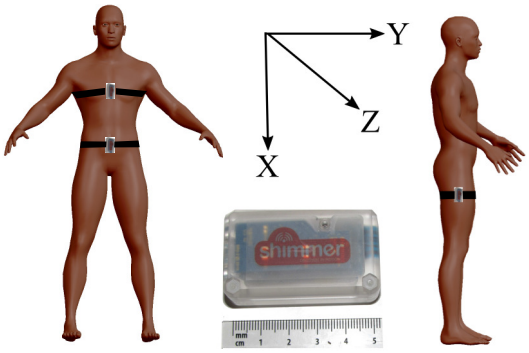


Figure 3.1: Sensor placements (left and right) and the shimmer sensor (middle) for the Cogent dataset

samples. Those numbers show that this dataset has a larger number of samples than both Noury *et al.* [115] (600 data points for both falls and ADLs) and Abbate *et al.* [2] (86 fall-like samples with 44 falls included). Although Noury *et al.*'s study provides more complex activity scenarios, their data are not publicly accessible. Table 3.2 shows the type of falls and ADLs (followed by their numbers of samples) while Figure 3.2 shows the staged falls and some ADLs of this dataset. More detailed information on the protocol can be found in Ojetola *et al.* [119].

3.2 SisFall dataset

3.2.1 Subject profile

This dataset has two groups of subjects: young adults and older people. This thesis uses the data from the young-adult group because the older-people group does not involve any fall activities. This young-adult group consists of 10 males and 11 females (age 25.0±8.6 years, height 165.7±9.3 cm, and weight 57.7±15.5 kg). This dataset has 1 subject aged 60. The number of subjects used in this thesis is fewer than the published dataset, as some subjects were removed due to incomplete samples.

Figure 3.2: ADLs and staged falls for the Cogent dataset [118]

Table 3.2: Types of falls and ADLs (together with their counts) in the Cogent dataset

Category	Activity	Number of events
ADLs	Standing while doing some other activities (e.g. making a phone call)	184
	Sitting on a chair while doing some other activities (e.g. reading a book)	184
	Near fall	276
	Sitting on the floor (not a result of falling)	276
	Lying on a bed while doing some other activities (e.g. reading a book)	92
	Walking while doing some other activities (e.g. making a phone call)	184
Falls	Forward	276
	Backward	185
	Left-side	91
	Right-side	92

3.2.2 Hardware

To collect the data, Sucerquia *et al.* [140] used a custom-made device that consists of a Kinetis MKL25Z128VLK4 microcontroller (NPX, Austin, Texas, USA), an Analog Devices (Norwood, Massachusetts, USA) ADXL345 accelerometer, a Freescale MMA8451Q accelerometer, an ITG3200 gyroscope, an SD card for recording, and a 1000 mAh generic battery. The device was placed on the subjects' waists. This thesis uses only data gathered from the ADXL345 accelerometer.

3.2.3 Protocol

Each subject performed 15 types of falls, each five times, and 19 types of ADLs. Thus, this dataset has 1,575 fall and 1,659 ADL data points in total. Table 3.3 shows the type of falls and ADLs (followed by their number of instances) in this dataset. The length of all fall events in this dataset is 15 seconds (uniform), while for the ADLs events the lengths are 12, 25, 100 seconds. Fall events are started with an activity such as walking, jogging, or sitting.

Table 3.3: Types of fall and ADLs (together with their counts) in the SisFall dataset

Category	Activity	Number of events
ADLs	Walking slowly (D01),	21
	Walking quickly (D02),	21
	Jogging slowly (D03),	21
	Jogging quickly (D04),	21
	Walking upstairs and downstairs slowly (D05),	105
	Walking upstairs and downstairs quickly (D06),	105
	Slowly sitting in a half-height chair, waiting a moment, and standing up slowly (D76),	105
	Quickly sitting in a half-height chair, waiting a moment, and standing up quickly (D08),	105
	Slowly sitting in a low-height chair, waiting a moment, and standing up slowly (D09),	105
	Quickly sitting in a low-height chair, waiting a moment, and standing up quickly (D10),	105
	Sitting a moment, trying to get up, and collapsing into a chair (D11),	105
	Sitting a moment, lying down slowly, waiting a moment, and sitting up again (D12),	105
	Sitting a moment, lying down quickly, waiting a moment, and sitting up again (D13),	105
	Being on one's back, changing to lateral position, waiting a moment, and changing to one's back (D14),	105
	Standing, slowly bending at knees, and getting up (D15),	105
	Standing, slowly bending without bending knees, and getting up (D16),	105
	Standing, getting into a car, remaining seated then getting out of the car (D17),	105
	Stumbling while walking (D18),	105
	Gently jumping without falling, while trying to reach a high object (D19).	105
Falls	Falling forward while walking caused by a slip (F01),	105
	Falling backward while walking caused by a slip (F02),	105
	Falling laterally while walking caused by a slip (F03),	105
	Falling forward while walking caused by a trip (F04),	105
	Falling forward while jogging caused by a trip (F05),	105
	Falling vertically while walking caused by fainting (F06),	105
	Falling while walking, with use of hands on a table to dampen fall, caused by fainting (F07),	105
	Falling forward when trying to get up (F08),	105
	Falling laterally when trying to get up (F09),	105
	Falling forward when trying to sit down (F10),	105
	Falling backward when trying to sit down (F11),	105
	Falling laterally when trying to sit down (F12),	105
	Falling forward while sitting, caused by fainting or falling asleep (F13),	105
	Falling backward while sitting, caused by fainting or falling asleep (F14),	105
	Falling laterally while sitting, caused by fainting or falling asleep (F15).	105

3.3 FARSEEING dataset

This dataset consists of 22 older subjects, where those subjects experienced real falls. Table 3.4 shows the body profiles of the subjects of this dataset. This dataset is available on request from the FARSEEING project [1]. Some subjects had sensors attached on their sacrum near L5, while the others had a sensor attached on their thigh. However, no subject had sensors attached on multiple body parts.

Regarding the hardware specification, ActivPal3[4] were used as the thigh sensor with a 20 Hz sampling rate. For the L5 sensor, a MiniMod[5] device was placed on the L5 segment of some of the subjects. For some subjects, a hybrid device is used as the L5 sensor. However, more detailed information regarding this hybrid device is not provided. This dataset has signals from an accelerometer (ms^{-2}), a gyroscope (°/s), and a magnetometer (μT). For this thesis, only signals from the accelerometer are used. This study focuses on using just accelerometer sensors (see subsection 2.5.1). Because the FARSEEING dataset uses ms^{-2} as its unit while the other datasets use g as their unit, a data conversion was done on the FARSEEING dataset by assuming $1g = 9.8\ ms^{-2}$.

This dataset has the following fall types: backward, forward, side forward, and backward on the left. Some activities were reported before the fall: walking, standing, and bending down. Falls are labeled as a single point in this dataset, thus they do not have a length. More detailed information about the FARSEEING dataset can be found in Appendix A.

3.4 Discussion

Bagala *et al.* [10] show that using data from younger subjects to determine thresholds to detect falls in older subjects is not effective, since some fall phases that are de-

[4]ActivPal is a product of the PALtechnologies company (http://www.paltechnologies.com/)
[5]MiniMod is a product of the McRoberts company (https://www.mcroberts.nl/)

Table 3.4: Subjects' body profiles of the FARSEEING dataset

Profile	**Subjects with L5 sensor**	**Subjects with thigh sensor**
Number of females	10	2
Number of males	5	5
Height (cm)	164.3±9.6	173.3±14
Weight (kg)	76.7±10.0	73.6±19.8
Age (years)	66.8±6.3	75.3±7.7

tected in younger subjects does not exist in acceleration signals from real falls that are experienced by older subjects. In contrast to Bagala *et al.*' study, Kangas *et al.* [85] and Jamsa *et al.* [82] show that data from younger subjects share a similar pattern to data from real falls of older subjects. Based on Ojetola's study [118] (this study uses data from younger subjects), during the pre-impact stage, the acceleration drops below $1g$ since the subject is (briefly) in a weightless state after losing their balance. This is followed by the impact stage where several high acceleration peaks occur as an indicator of the moment when the subject's body hits the ground. An inactive state is shown after the impact stage, where this condition is a characteristic of the post-impact stage. These fall stages are shown to exist on the staged-fall acceleration signal on simulated/staged falls from the Cogent and SisFall datasets (Figures 3.3 and 3.4) and the real-fall acceleration signal from the FARSEEING dataset (Figure 3.5). This finding can be an indication that real falls have a similar pattern to laboratory-based falls in terms of the fall stages. Thus, to be precise, subsection 5.4.7 provides a performance comparison between using data from young subjects (the Cogent and SisFall datasets) and older subjects (FARSEEING dataset).

Another issue that arises from the laboratory-based dataset is the use of a mattress during a data collection, by the Cogent and SisFall datasets. Kangas *et al.* [85] show that some real falls produce higher impacts than staged falls, because the

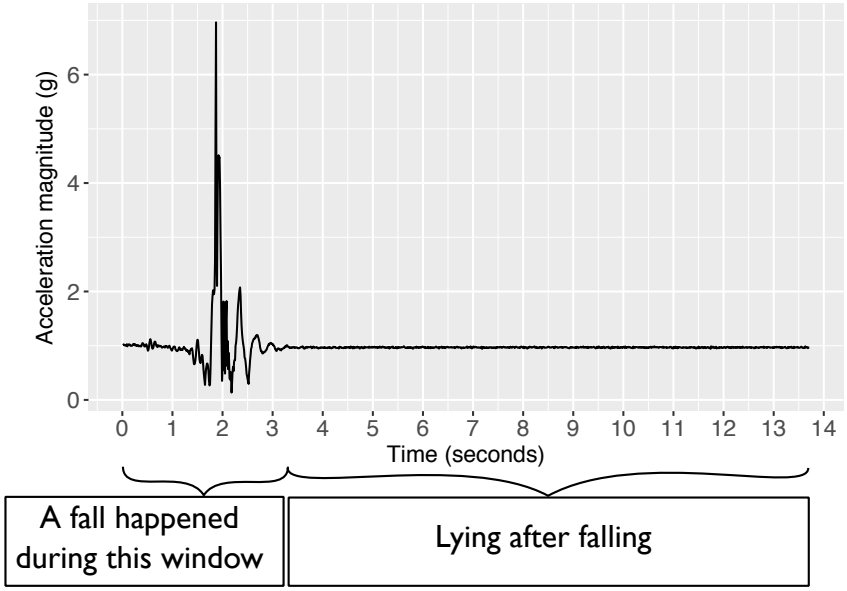


Figure 3.3: Fall acceleration magnitude (g) from the Cogent dataset

staged falls use a mattress during the data collection. Based on Klenk *et al.*'s study [96], using a mattress when performing a fall can damp the impact signal. However, their study does not investigate the effect of using a mattress on the data. Also, based on Casilari *et al.* [33], there is not yet a study investigating the impact of using a mattress on collecting data for a fall-detection study. Thus, this issue is still open for future work.

3.5 Summary

This chapter reviews the publicly-accessible datasets in this study: Cogent, SisFall, and FARSEEING. The aim of using publicly-accessible datasets is to get a fair comparison between techniques. The data were mostly gathered from young and healthy subjects (there is one subject whose age is 60 in the SisFall dataset) for the Cogent and SisFall datasets, while the FARSEEING dataset gathered data from older people. For the Cogent and SisFall datasets, their subjects staged some falls and activities of daily living (ADLs) in a laboratory environment. The results of this study still can be an indication of the performance of the fall-detection technique in

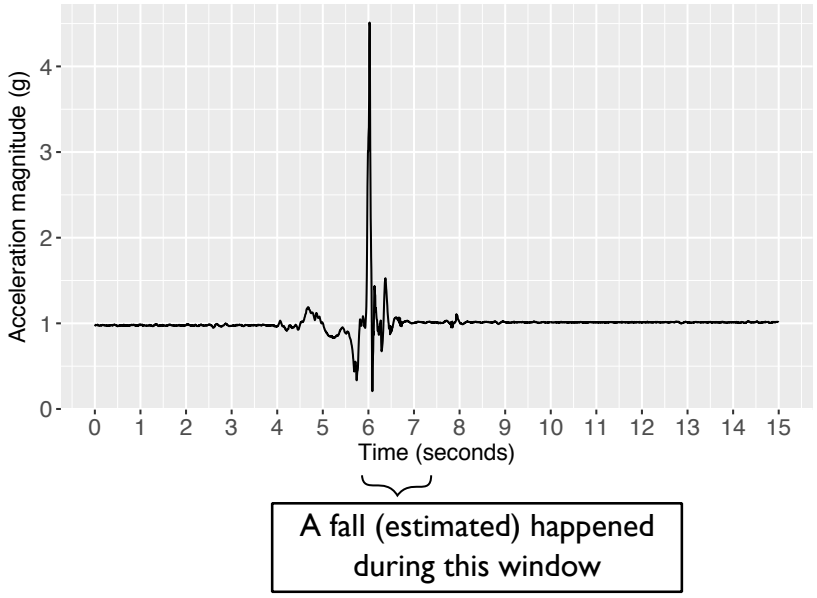


Figure 3.4: Fall acceleration magnitude (g) from the SisFall dataset

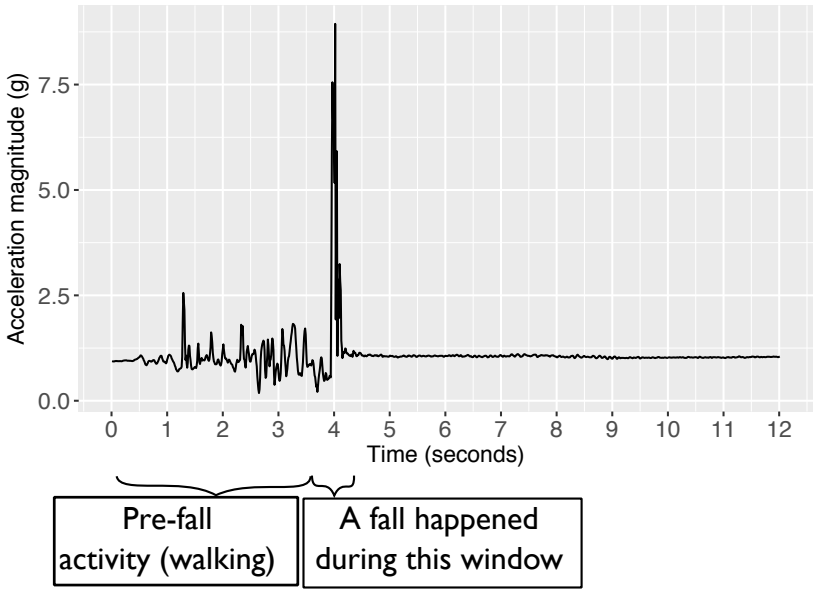


Figure 3.5: Fall acceleration magnitude (g) from the FARSEEING dataset

real-world cases, because the laboratory-based falls (the Cogent and SisFall datasets) share similar patterns with real falls from older people (the FARSEEING dataset). The next chapter provides an analysis of the performance of the current fall-detection approaches using the Cogent and SisFall datasets.

Chapter 4

An analysis of fall-detection approaches

4.1 Introduction

The previous chapter discusses three publicly-accessible datasets: Cogent, SisFall, and FARSEEING. This chapter focuses on analysing threshold- and sliding-window-based machine-learning approaches on these publicly accessible datasets. This chapter covers: an analysis of the impact of the window and overlap sizes of the sliding window for machine-learning-based approaches, and a comparison between sliding-window-based machine-learning and threshold-based approaches, using publicly accessible datasets.

Machine-learning-based approaches (to do both learning and testing from a data stream) need to segment the data sequence and extract features from each segment. For traditional machine-learning-based fall-detection approaches, the segmentation techniques are categorised into two: Fixed-size Non-overlapping Sliding Window (FNSW) [50, 148] and Fixed-size Overlapping Sliding Window (FOSW) [43, 44, 118]. Although studies have been done for machine-learning based approaches, there is a

lack of information regarding the impact of the window size for the FNSW-based machine-learning approach and the overlap size for FOSW-based machine-learning approaches. Therefore, this chapter focuses on investigating the impact of window and overlap sizes on the classifier's detection rate (precision, recall, and F-score) using two publicly accessible datasets: Cogent and SisFall (see Chapter 3 for detailed information about the datasets).

A study from Vallejo *et al.* [148] shows that defining thresholds to detect falls is difficult, because there are data overlaps between falls and activities of daily living (ADLs). However, their study does not compare the detection rate of threshold- and machine-learning-based approaches. Azis *et al.* [8] show that using machine learning to build the classifier can give a better sensitivity and specificity than using pre-defined thresholds. However, their dataset is not publicly accessible. Thus, this chapter also covers a detection-rate comparison between threshold- and sliding-window-based machine-learning approaches using publicly accessible datasets (Cogent and SisFall), where these datasets have a relatively large number of subjects and type of activities.

The structure of this chapter: Section 4.2 provides the methodology of this chapter. Sections 4.3 and 4.4 discus the detection rate of threshold- and machine-learning-based approaches, respectively. Section 4.5 discusses and analyses the results of the experiments on both threshold- and machine-learning-based approaches and the limitations of this chapter. A chapter summary is provided in Section 4.6.

4.2 Method

This chapter uses the Cogent and SisFall datasets. Three evaluations were conducted: an evaluation of a threshold-based approach, an evaluation of machine-learning-based approaches, and a comparison between a threshold-based approach and machine-learning-based approaches. A threshold-based approach called IM-

PACT+POSTURE from Kangas *et al.* [86] is used, because this algorithm is simple and provides a relatively high accuracy. Although this technique is relatively old in terms of time of publication (Kangas *et al.* published their paper in 2008), it is still used as a comparison in Aziz *et al.*'s [8] study. In fact, this chapter shows that this simple and relatively old technique can achieve similar performance to machine-learning-based approaches when the Cogent dataset is used. All thresholds are determined based on the dataset using an approach from Kangas *et al.* [87].

In this chapter, although recorded datasets were used, the experiment was implemented in a real-time-style simulation where the sample appears one by one. The FNSW and FOSW techniques were implemented from the beginning of the record. This means that the window does not necessarily centred on the acceleration peak. An illustration of the real-time-style simulation is shown in Figure 4.1. Both FNSW- and FOSW-based machine-learning approaches use several machine-learning algorithms in their implementation. For training and testing the classifier, four machine-learning algorithms from the Scikit-learn library [125] were used: Classification and Regression Tree (CART), Logistic Regression (LR), Support Vector Machine (SVM), and k-Nearest Neighbour (k-NN). The following parameters are used for the machine-learning algorithms:

- $k = 1, 2$, and 3 together with the Euclidean distance for k-NN;
- an inverse of the regularisation strength (C) of $10^8, 10^9$, and 10^{10} for LR;
- linear, radial basis function (RBF), and polynomial (with $d = 3$, where this value is the default value of the Scikit-learn library) kernels for SVM.

The Cogent dataset shows that falls mostly last 2 seconds or longer. Figure 4.2 shows the length of falls from the Cogent dataset. This length is obtained based on the label of the fall event. Thus, 2 to 12 s was used as the range of used size to assess the Cogent dataset, as the largest window size from current fall-detection studies is 12 seconds [118]. For the SisFall dataset, this thesis considers windows

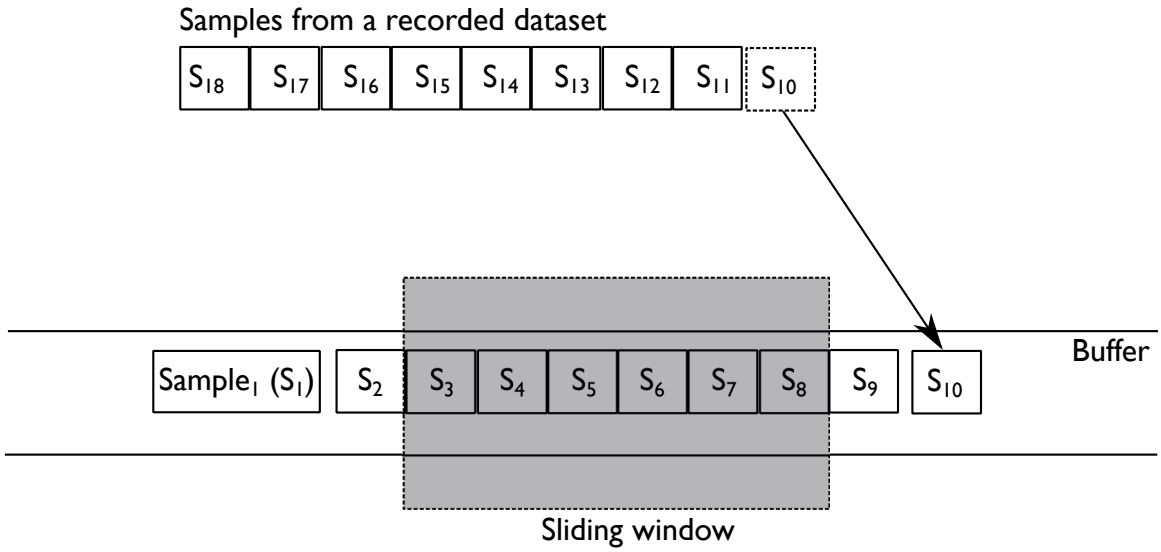


Figure 4.1: An illustration of the use of a sliding window in a real-time-style simulation

with lengths of 2 to 15 s, because the length of all fall events from this dataset is 15 s (uniform). The FOSW-based machine-learning approaches implement several window overlaps: 25%, 50%, 75%, and 90% [19].

This study uses leave-one-subject-out cross-validation (LOSOCV) as the classifier evaluation method. For the threshold-based approach, LOSOCV means using $N - 1$ subjects (N is the total number of subjects) to adjust thresholds and using one subject as a test case. That process is repeated until all subjects have been used as a test case in turn. LOSOCV for the machine-learning-based approach means using $N - 1$ subjects (N is the total number of subjects) to train a classifier using a machine-learning algorithm, and using one subject as a test case. Then, this validation technique does iterations until all subjects have been used as a test case.

As the number of fall data is very small compared to those of ADLs, accuracy cannot be used to measure the classifier's performance because it overvalues the always-negative classifier [55]. Thus, this study uses precision, recall, and F-score. When a sequence (annotated as a particular activity) is segmented for online processing by a fall-detection approach (e.g. a threshold-based approach or an (FNSW- or FOSW-based) machine-learning approach), several segments are usually produced

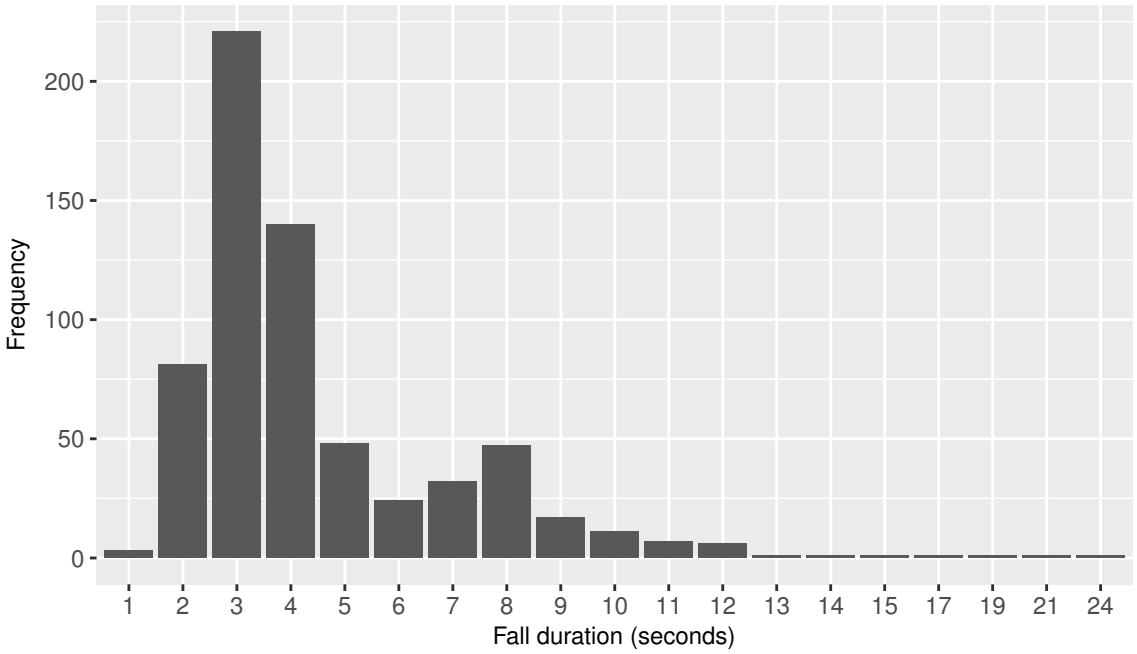


Figure 4.2: The length of fall events from the Cogent dataset (this length is defined based on the annotation of the data)

from that particular activity. Since the classifier does the classification on each segment, those segments might produce different results (Figure 4.3). To obtain a single classification result, the following rules are implemented:

- A data sequence is detected as an FP if this data sequence is annotated as a non-fall activity and at least one segment out of all the segments produced from this data sequence is detected as a fall.
- A data sequence is detected as a TP if this data sequence is annotated as a fall activity and at least one segment out of all the segments produced from this data sequence is detected as a fall.
- A data sequence is detected as an FN if this data sequence is annotated as a fall activity and no segment is detected as a fall.

At a certain time, a segment may include samples from both fall and non-fall activities. This segment is annotated as a fall if it has samples from a fall activity. Note that the total number of segments that are produced by an activity varies depends

on the length of the activity, the length of sliding window, and the type of sliding window (FNSW or FOSW). A Wilcoxon signed-rank test was used to evaluate the significance of the improvement in the detection rate. This method is chosen because the precision, recall, and F-score values are not normally distributed. This study uses the Shapiro-Wilk normality test to assess the distribution of precision, recall, and F-score values.

4.3 Threshold-based fall-detection approach

The main idea of a threshold-based fall-detection approach is using manually predefined thresholds to distinguish falls from ADLs. This chapter implements a threshold-based approach from Kangas *et al.* [86] called IMPACT+POSTURE.

4.3.1 IMPACT+POSTURE approach

IMPACT+POSTURE detects falls by finding an impact followed by monitoring the posture of the subject's body. To find an impact, several thresholds are used:

- Total sum vector (SV_{tot}). First of all, this approach filters the accelerometer signal using a median filter with a window length of three samples to reduce noise. Then SV_{tot} is calculated using

$$SV_{tot} = \sqrt{(A_x)^2 + (A_y)^2 + (A_z)^2}, \tag{4.1}$$

where A_x, A_y, and A_z are the accelerations (g) of the x-, y-, and z-axes, respectively. This parameter contains both dynamic and static acceleration components.

- Dynamic sum vector (SV_D). In the beginning, this approach filters the accelerometer signal using a median filter with a window length of three samples, followed by a high-pass filter ($f_c = 0.25$ Hz) using a digital second-order But-

(a) Using the Cogent dataset

(b) Using the SisFall dataset

Figure 4.3: The real-time classification process using a sliding-window technique

terworth filter. After that, formula (4.1) is used to calculate SV_D as it can detect fall-related impacts.

- The difference between the maximum and the minimum of the total sum vector (SV_{maxmin}) is calculated by constructing a sliding sum vector, generated by calculating the difference between the maximum and the minimum values of the total sum vector in a 0.1-second sliding window for each axis.

- Vertical acceleration (Z_2). This approach calculates this parameter using

$$Z_2 = \frac{SV_{tot}^2 - SV_d^2 - G^2}{2G}, \tag{4.2}$$

where G is the gravitational acceleration ($G = 1g$).

If one of the parameters above exceeds the thresholds, the approach measures the posture of the subject's body. The thresholds are calculated using a technique from Kangas *et al.* [87]. To avoid biased results, the thresholds are defined from the training set and they are evaluated on the test set. Figure 4.4 shows the evaluation steps of the IMPACT+POSTURE algorithm using LOSOCV. To calculate posture, similarly to the previous parameters, the approach filters the signal from the z-axis using a median filter with a window length of three samples [86, 90]. Then the data is low-pass filtered ($f_c = 0.25$ Hz) using a digital second-order Butterworth filter. Two seconds after the impact, the approach gathers samples in a 0.4 s time interval for posture detection. If the average of the samples in that interval is equal to or lower than $0.5g$, the approach detects this data as a lying posture [86, 90]. Because this parameter relies on the signal from the z-axis, the sensor position needs to be unchanged during the data collection. In fact, for both the Cogent and SisFall datasets, the sensors were strapped on the subject's body, which means that the position of the sensor is unlikely to have changed.

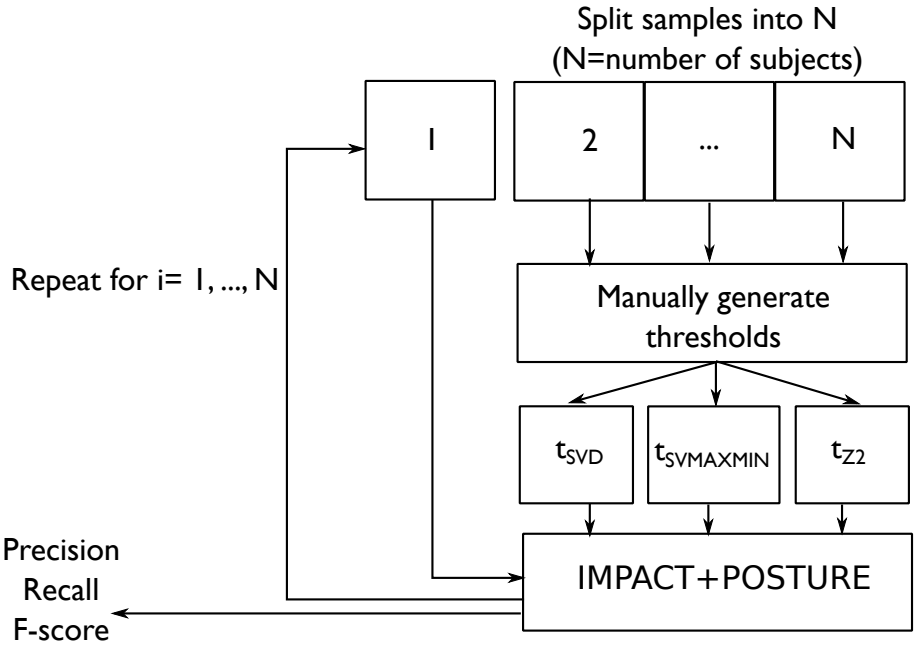
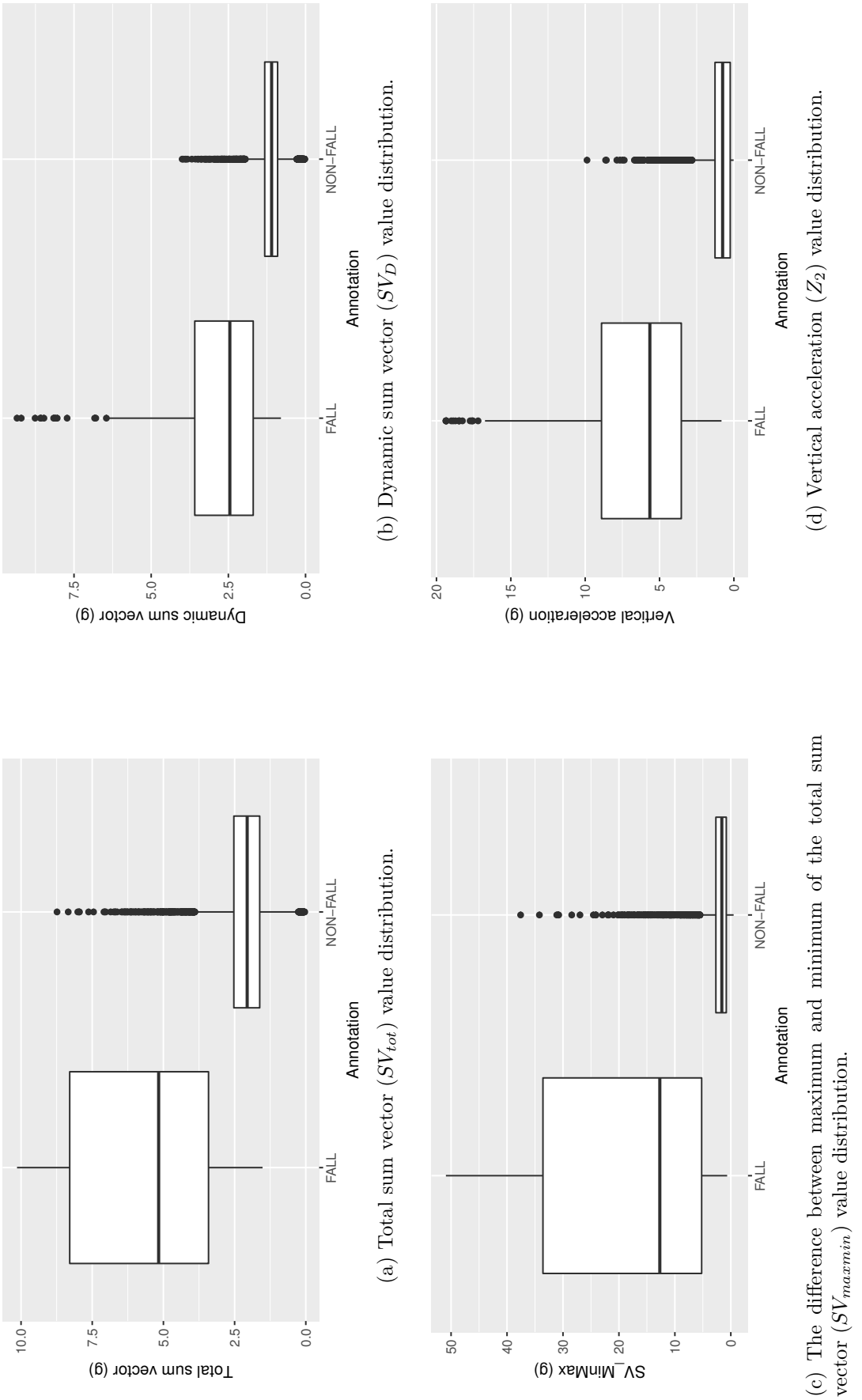


Figure 4.4: Evaluation of IMPACT+POSTURE algorithm using leave-one-subject-out cross-validation (LOSOCV)

4.3.2 Performance analysis

4.3.2.1 Cogent dataset

Figure 4.5 shows the distributions of all the parameters (SV_{tot}, SV_D, SV_{maxmin}, and Z_2) extracted from the Cogent dataset. Figures 4.5a–4.5d indicate that there are data overlaps between falls (FALL) and ADLs (NON-FALL). The data overlap means an overlap between the feature values of fall and non-fall events in the feature space. These overlaps can cause both false alarms (false positive) and undetected falls (false negatives). Instances of NON-FALL that exceed the threshold can cause false alarms, while instances of FALL that have values below the threshold can cause undetected falls. For this study, the lowest value from falls is defined as the threshold, aiming to detect all falls. Table 4.1 shows the detection rate, in terms of precision, recall, and F-score, of IMPACT+POSTURE. In general, it can be seen that IMPACT+POSTURE can achieve up to 88.6% for the F-score on average.



(a) Total sum vector (SV_{tot}) value distribution.

(b) Dynamic sum vector (SV_D) value distribution.

(c) The difference between maximum and minimum of the total sum vector (SV_{maxmin}) value distribution.

(d) Vertical acceleration (Z_2) value distribution.

Figure 4.5: Parameter distributions of the Cogent dataset

Table 4.1: IMPACT+POSTURE performance

Metric	Cogent dataset (%)	SisFall dataset (%)
Precision	90.9±10.2	54.1±1.7
Recall	87.6±11.9	100±0
F-Score	88.6±9.2	70.2±1.4

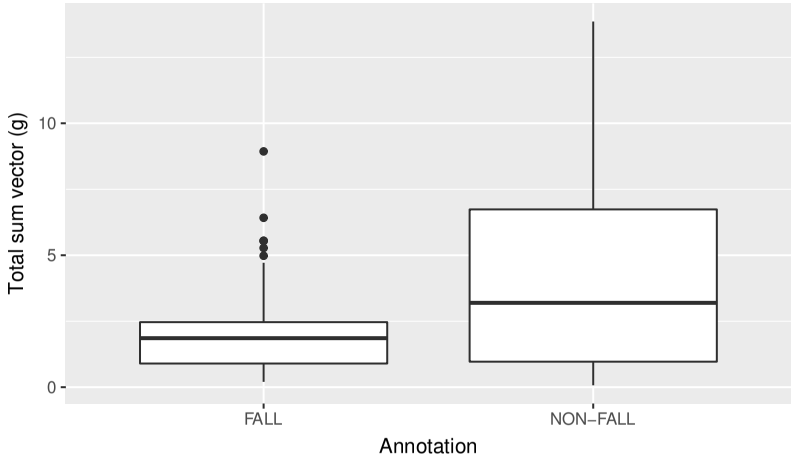
4.3.2.2 SisFall dataset

Figure 4.6 shows the distribution of all the parameters (SV_{tot}, SV_D, SV_{maxmin}, and Z_2) extracted from the SisFall dataset, while Table 4.1 shows the performance of IMPACT+POSTURE tested on the SisFall dataset. Although all falls from the SisFall dataset are correctly detected (100% recall) by IMPACT+POSTURE, the number of false alarms is relatively high, as can be seen from the classifier's precision. The precision achieved is very poor, at 54.1% on average. Overall, IMPACT+POSTURE is able to achieve a 70.2% F-score on average.

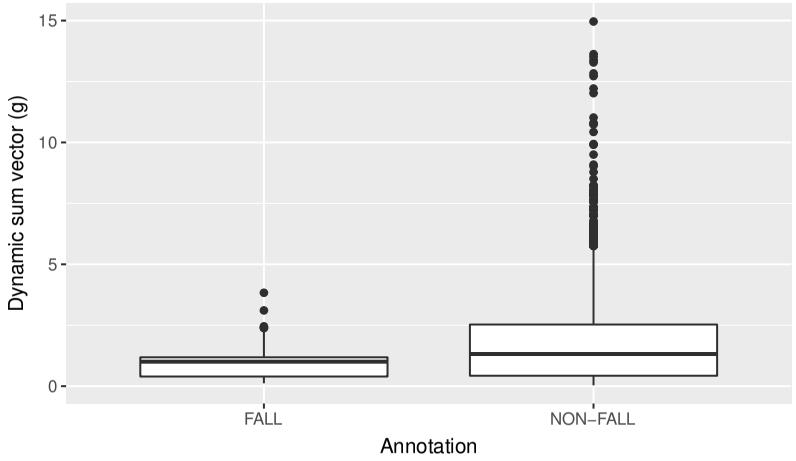
A lesson learned from this section is that manually defining the threshold [25, 34, 47, 87, 86, 88, 136, 140] is not a trivial task, and might cause the number of undetected falls or the number of false alarms to increase. Sub-section 4.4 investigates existing machine-learning-based approaches for fall detection.

4.4 Machine-learning-based fall-detection approach

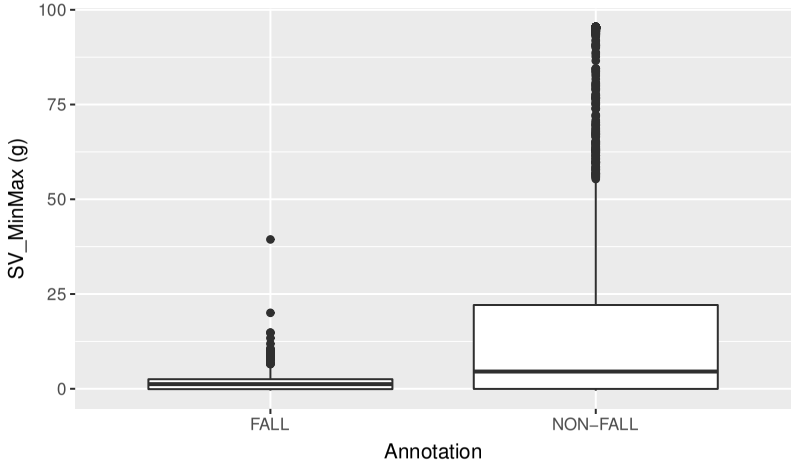
Most of the machine-learning-based approaches use FNSW or FOSW to segment the data stream before doing a feature-extraction process (see subsection 2.3.2). This sub-section investigates the use of FNSW and FOSW on several machine-learning algorithms (CART, k-NN, LR, and SVM). The following features are used in this study:



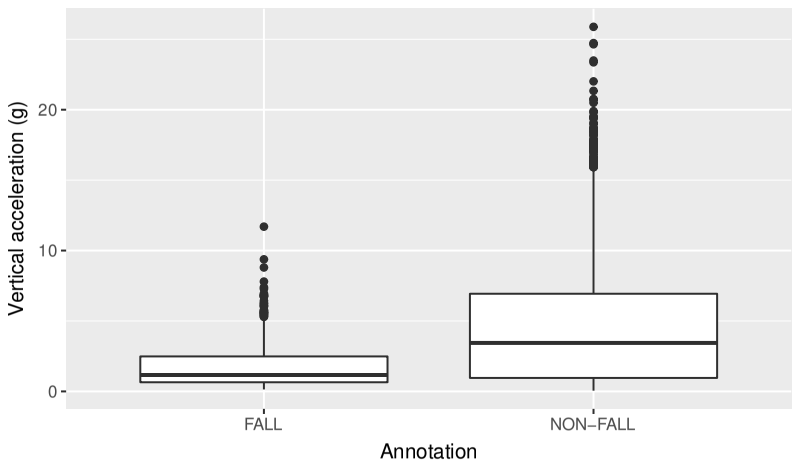
(a) Total sum vector (SV_{tot}) value distribution.



(b) Dynamic sum vector (SV_D) value distribution.



(c) The difference between maximum and minimum of the total sum vector (SV_{maxmin}) value distribution.



(d) Vertical acceleration (Z_2) value distribution.

Figure 4.6: Parameter distribution of the SisFall dataset

1. Minimum, maximum, and average acceleration-vector magnitudes [2, 118, 127].

2. Velocity:

$$\Delta V = \frac{1}{f} \sum_{m=1}^{n} a_m, \tag{4.3}$$

where a_m and f are the acceleration-vector magnitude and sampling frequency, respectively [24, 118, 127].

3. Energy expenditure [39, 108, 118, 157], where the energy expenditure is related to the acceleration signal (with 89% correlation based on Mathie *et al.* [108]) by:

$$E = \alpha \left(\sum_{i=1}^{n} a_x^2 + \sum_{i=1}^{n} a_y^2 + \sum_{i=1}^{n} a_z^2 \right),$$

where α is a constant of proportionality, where in this study $\alpha = 1$. The term energy expenditure is used in a loose sense – it does not necessarily correspond to the expended kinetic energy. Nonetheless, it is a useful feature that is easy to calculate.

4. Variance of the acceleration-vector magnitude [29, 118, 127].

5. Root mean square (RMS) of the acceleration-vector magnitude [24, 64, 127].

6. Acceleration exponential moving average (EMA) [29, 118, 127]:

$$s_t = \alpha V_m + (1 - \alpha) s_{t-1},$$

where s_t and V_m are the EMA value and acceleration-vector magnitude, respectively.

7. Signal-magnitude area (SMA) [90, 118, 127, 164]:

$$\gamma = \frac{1}{n} \left(\sum_{i=1}^{n} |a_{ix}| + \sum_{i=1}^{n} |a_{iy}| + \sum_{i=1}^{n} |a_{iz}| \right),$$

Variables a_x, a_y, a_z, and n from the formulas above are the outputs of the accelerometer on the x, y, z axes, and the number of samples in a segment, respectively.

4.4.1 Classification and Regression Tree (CART)-based fall-detection approach

4.4.1.1 Sliding-window+CART performance on the Cogent dataset

Figure 4.7 shows the impact of the window size on the precision, recall, and F-score of the FNSW and FOSW with a CART-based classifier tested on the Cogent dataset. The FNSW+CART-based classifier achieves a better precision when the window size is increased. On the other hand, to get a better recall, a smaller window is needed by the classifier. This means that using a larger window might increase the number of undetected falls while a smaller window increases the number of false positives. Overall, using the window size of 12 seconds gives 64.8±12.7%, 87±15.2%, and 73.6±12.2% of precision, recall, and F-score, respectively.

Table 4.2 shows the overall performance of the FOSW+CART-based classifier on its window overlaps (25%, 50%, 75%, and 90%) in terms of precision, recall, and F-score. From this table, it can be seen that increasing the data overlap can increase the recall to 99.1% on average. However, the precision decreases when the data overlap increases. This means that increasing the data overlap can cause a classifier to increase its number of false alarms, while reducing its number of undetected falls. In terms of F-score and precision, increasing the overlap size can degrade the classifier's performance. In terms of the window overlap size, the classifier can achieve the best result when the overlap is 25%. The classifier can achieve the best overall result when the window size is 11 seconds with 25% of data overlap, with a 58.3±10.3% precision, an 89.8±13.1% recall, and a 70.2±9.8% F-score.

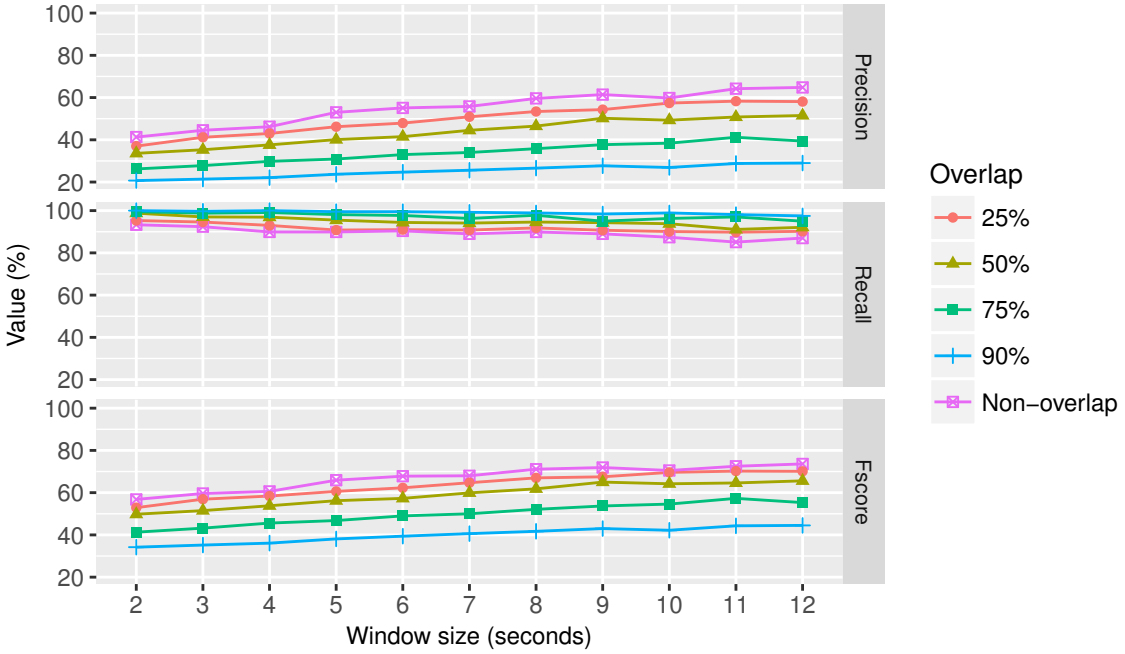


Figure 4.7: FOSW+CART precision, recall, and F-score (%) for different window overlap sizes (Cogent dataset)

Table 4.2: Overall FOSW+CART performance (average and standard deviation) based on the overlap size using the Cogent dataset

Overlap	**Metric (%)**		
size (%)	**Precision**	**Recall**	**F-score**
25	49.8±12.5	91.6±12	63.7±11.7
50	43.7±10.5	94.8±8.6	59.1±9.9
75	34.0±7.9	97.3±6.2	49.9±8.6
90	25.2±4.9	99.1±3.1	39.0±6.1

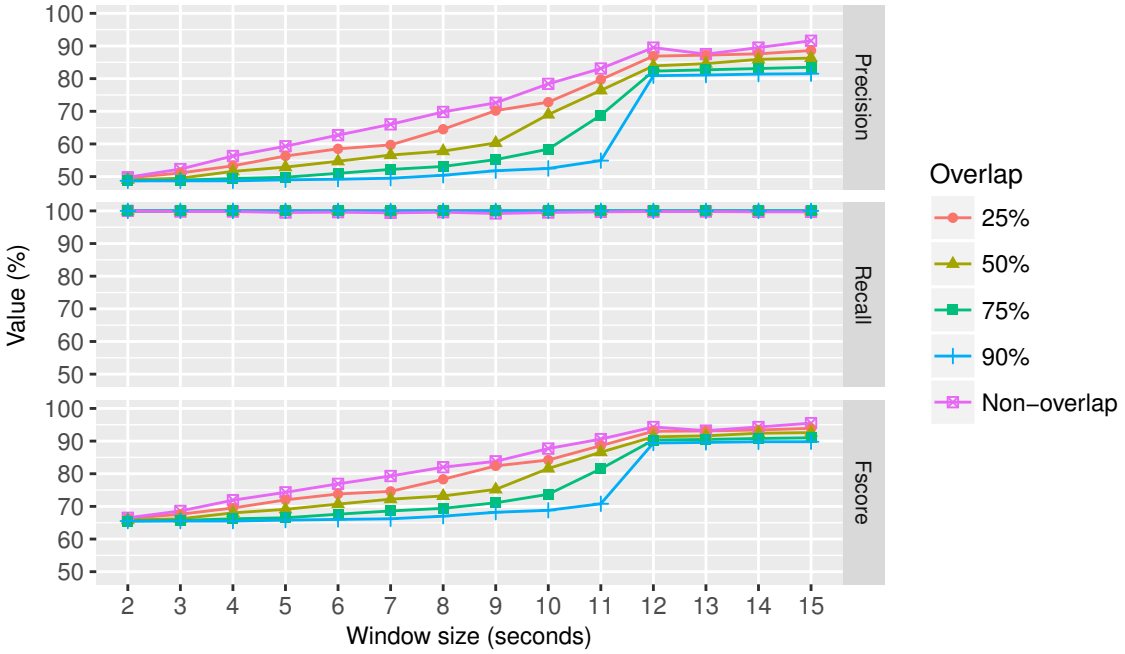


Figure 4.8: FOSW+CART precision, recall, and F-score (%) for different overlap sizes (SisFall dataset)

4.4.1.2 Sliding window+CART performance on the SisFall dataset

Figure 4.8 shows the impact of the window size on the precision, recall, and F-score of FNSW and FOSW with a CART-based classifier. The FNSW+CART-based classifier can get a better precision when the window size is increased. An increase in the size of the window can cause a slight reduction in recall. Overall, the classifier can achieve the best result when the window size is 15 seconds, with a 91.6±2.4% precision, a 99.7±0.7% recall, and a 95.5±1.5% F-score.

Table 4.3 shows the precision, recall, and F-score of FOSW with a CART-based classifier tested on the SisFall dataset. From these results, it can be seen that increasing the overlap of the FOSW can reduce the precision of the classifier. Although most of the overlaps can detect all falls, they produce a relatively low precision, which is a sign of an increase in false alarms. Overall, for the FOSW+CART approach, the classifier can achieve the best result when the overlap size is 25% and the window size is 15 seconds (see Figure 4.8), with 88.6±2.7%, 99.9±0.4%, and 93.9±1.5% of precision, recall, and F-score, respectively.

Table 4.3: FOSW+CART performance (average and standard deviation) based on the overlap size using the SisFall dataset

Overlap	**Metric (%)**		
size (%)	**Precision**	**Recall**	**F-score**
25	69.0±14.5	99.9±0.4	82.2±12.0
50	65.6±14.4	100±0.1	78.3±10.3
75	61.9±14.3	100±0.0	75.6±10.4
90	59.2±14.1	100±0.0	73.4±10.4

4.4.2 k-nearest neighbours (k-NN)-based fall-detection approach

4.4.2.1 Sliding window + k-NN performance on the Cogent dataset

Figure 4.9 shows the impact of the window size on the precision, recall, and F-score values of FNSW and FOSW with the k-NN machine-learning algorithm tested on the Cogent dataset. These results show that increasing the window size can cause the FNSW+k-NN-based classifier's precision to increase. On the other hand, the classifier's recall decreases when the window size is increased. In terms of F-score, the classifier's performance improves when a bigger window is used. Figure 4.10 shows the F-scores of FNSW+k-NN for different k values. The classifier is able to achieve the best result when $k = 3$ and the window size is 9 seconds, with an 83.9±10.8% precision, an 89.3±12.8% recall, and an 85.9±9.5% F-score.

Table 4.4 shows the performance (in terms of precision, recall, and F-score) using FOSW with k-NN on several window overlaps. Increasing the overlap size tends to decrease the precision. This means that increasing the window size of the FOSW+k-NN-based classifier can produce more false alarms. On the other hand, increasing the overlap size tends to increase the recall, which means that the number of undetected falls is reduced. In general, in terms of F-score, reducing the

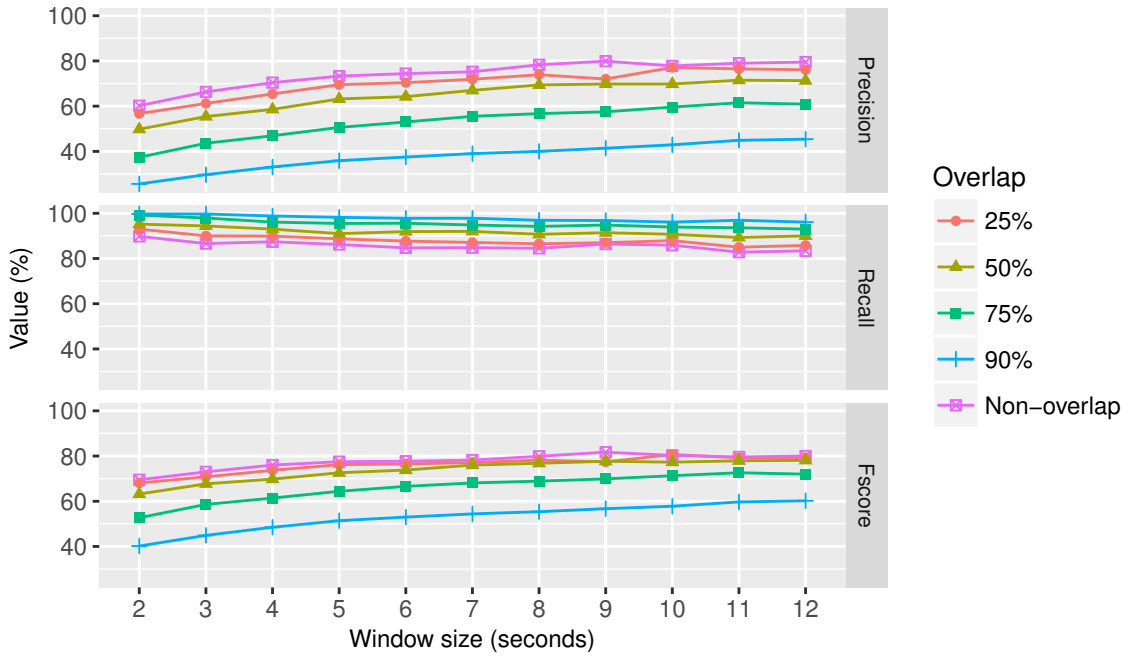


Figure 4.9: FOSW+k-NN precision, recall, and F-score (%) for different window and overlap sizes (Cogent dataset)

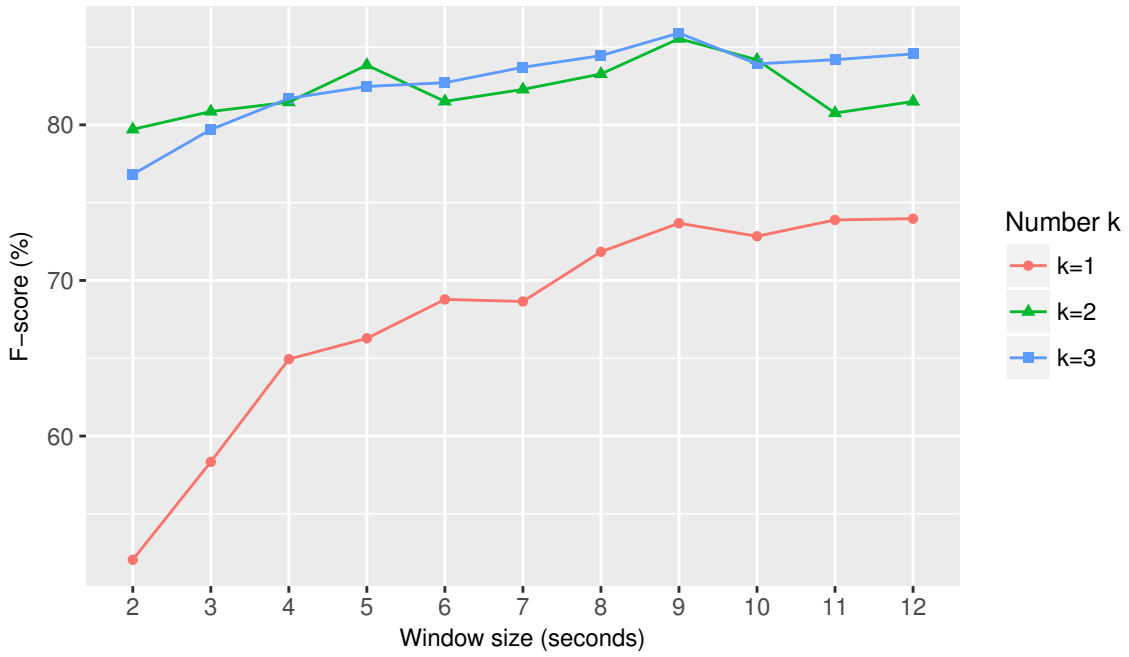


Figure 4.10: FNSW+k-NN F-score (%) for different k values (Cogent dataset)

Table 4.4: FOSW+k-NN overall performance (average and standard deviation) based on the overlap size using the Cogent dataset

Overlap	Metric (%)		
size (%)	Precision	Recall	F-score
25	70.1±19.1	88.1±15.1	76.1±15.0
50	64.5±19.6	91.8±12.1	73.7±15.2
75	53.0±18.8	95.3±9.3	66.0±16.0
90	37.7±14.3	97.7±6.3	52.9±14.3

data overlap size tends to degrade the classifier's performance. In terms of window overlap size, the classifier can achieve the best overall performance when the overlap size is 25%. Figure 4.11 shows the F-scores of FOSW+k-NN for different k values when the window overlap is 25%. Overall, the classifier can achieve the best result when $k = 3$, the window size is 10 seconds, and the overlap size is 25%, with an 82.3±12% precision, a 90.2±14.5% recall, and an 85.4±11.3% F-score.

4.4.2.2 Sliding window + k-NN performance on the SisFall dataset

Figure 4.12 shows the impact of the window size on the precision, recall, and F-score values of FNSW and FOSW with a k-NN-based classifier implemented on the SisFall dataset. These results show that the FNSW+k-NN-based classifier can achieve better precision and F-score when the window size increases. On the other hand, the recall decreases slightly when the window size increases. To find the best k for the classifier, Figure 4.13 shows the F-score of the classifier with different k values. The classifier is able to achieve the best result when the window size is 15 seconds and $k = 2$, with a 94.2±2.2% precision, a 98.9±1.3% recall, and a 96.5±1.4% F-score.

Table 4.5 shows the overall precision, recall, and F-score of FOSW from a k-NN-based classifier using the SisFall dataset. Similarly to the FOSW+CART-based

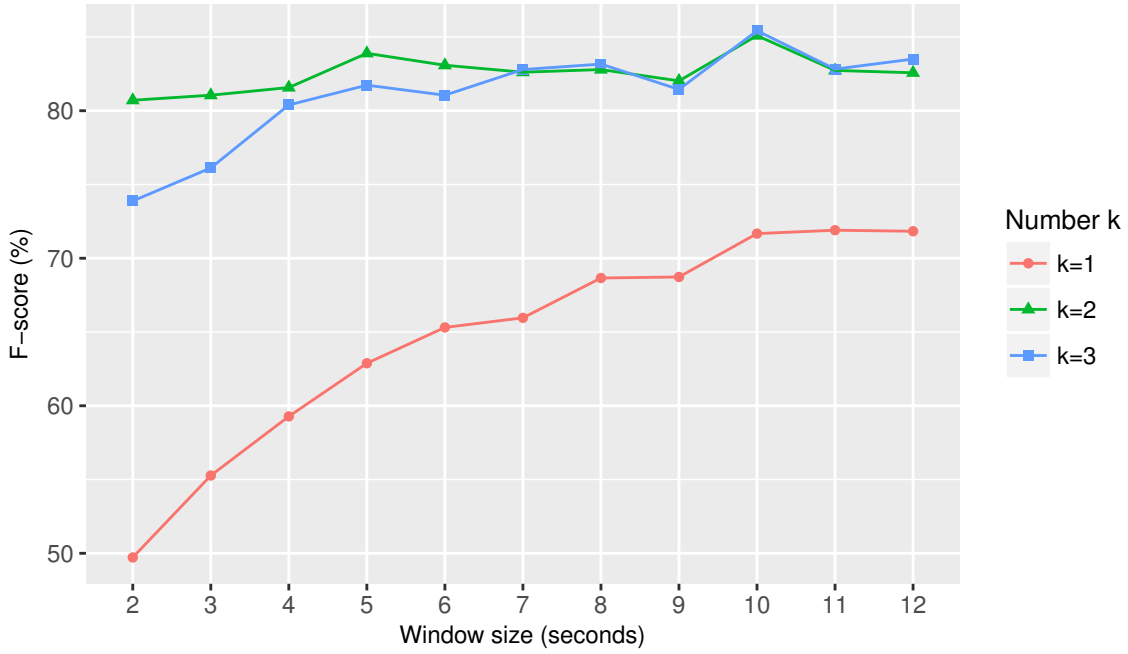


Figure 4.11: FOSW+k-NN F-score (%) for different k values when overlap size is 25% (Cogent dataset)

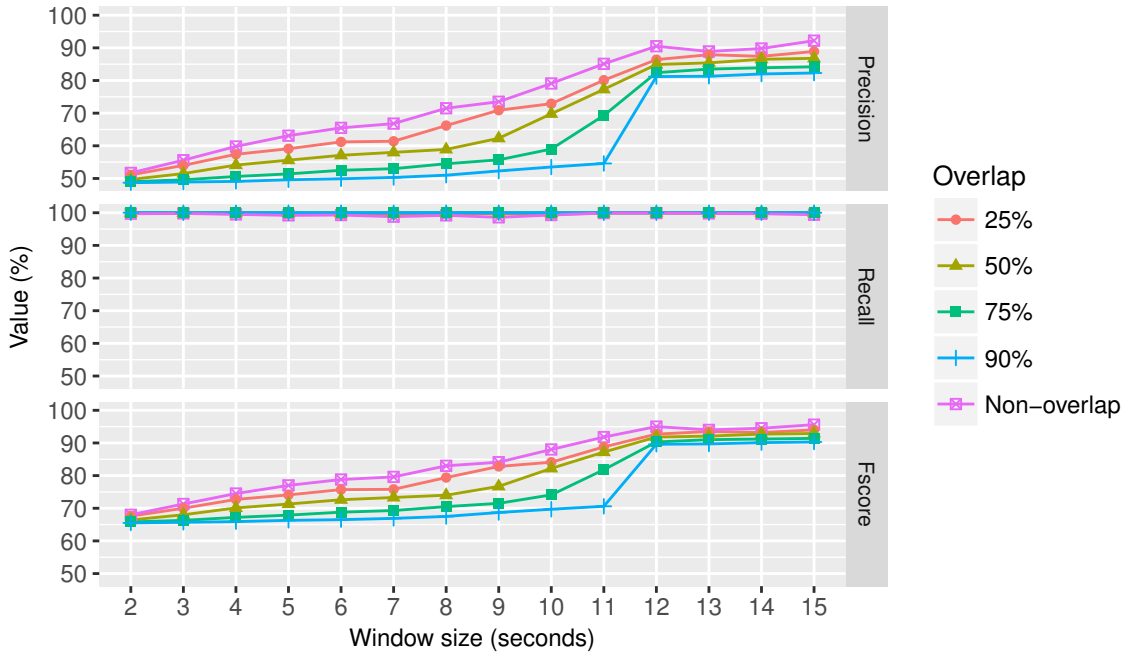


Figure 4.12: FOSW+k-NN precision, recall, and F-score (%) for different overlap sizes (SisFall dataset)

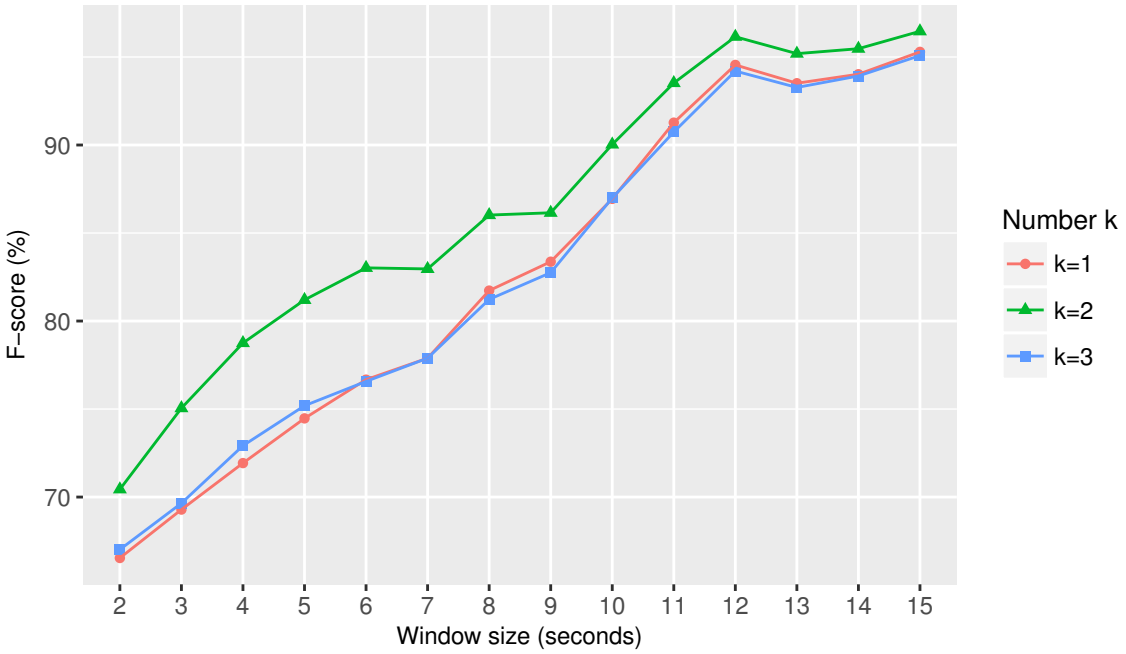


Figure 4.13: FNSW+k-NN F-score (%) for different k values (SisFall dataset)

classifier, increasing the window overlap does not necessarily improve the classifier's performance in general (in terms of F-score). The precision is decreased when the overlap size increases. This means that the number of false alarms increases when the window overlap increases. The FOSW+k-NN-based classifier achieves the best result when the window overlap is 25%, the window size is 15 seconds, and $k = 2$ (see Figure 4.14), with 91.4±2.6%, 99.6±0.9%, and 95.3±1.4% of precision, recall, and F-score, respectively.

4.4.3 Logistic-regression (LR)-based fall-detection approach

4.4.3.1 Sliding window + LR performance on the Cogent dataset

Figure 4.15 shows the impact of the window size on the performance (in terms of precision, recall, and F-score) of using FNSW and FOSW with an LR-based classifier. The precision tends to fluctuate when the window size increases for the FNSW+LR-based classifier. In terms of recall, increasing the window size tends to

Table 4.5: FOSW+k-NN overall performance (average and standard deviation) based on the overlap size using the SisFall dataset

Overlap	**Metric (%)**		
size (%)	**Precision**	**Recall**	**F-score**
25	70.4±13.8	99.7±0.8	81.7±9.5
50	67.0±14.2	100±0.2	79.4±10.0
75	62.8±14.2	100±0.0	76.2±10.3
90	59.6±14.2	100±0.0	73.8±10.4

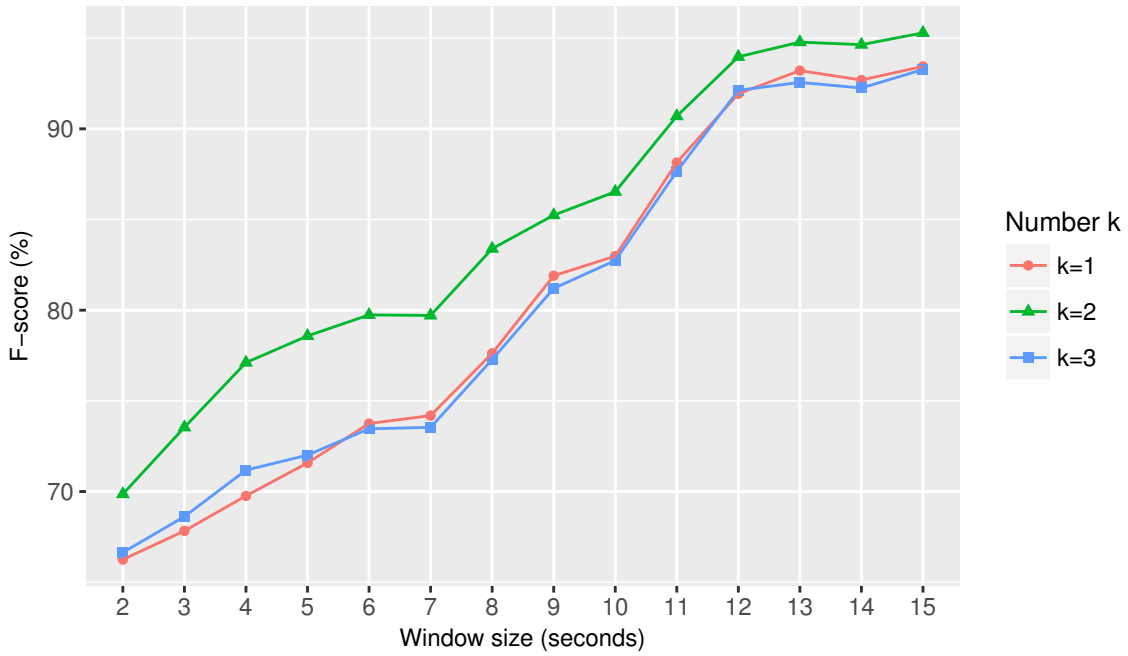


Figure 4.14: FOSW+k-NN F-score (%) for different k values when overlap size is 25% (SisFall dataset)

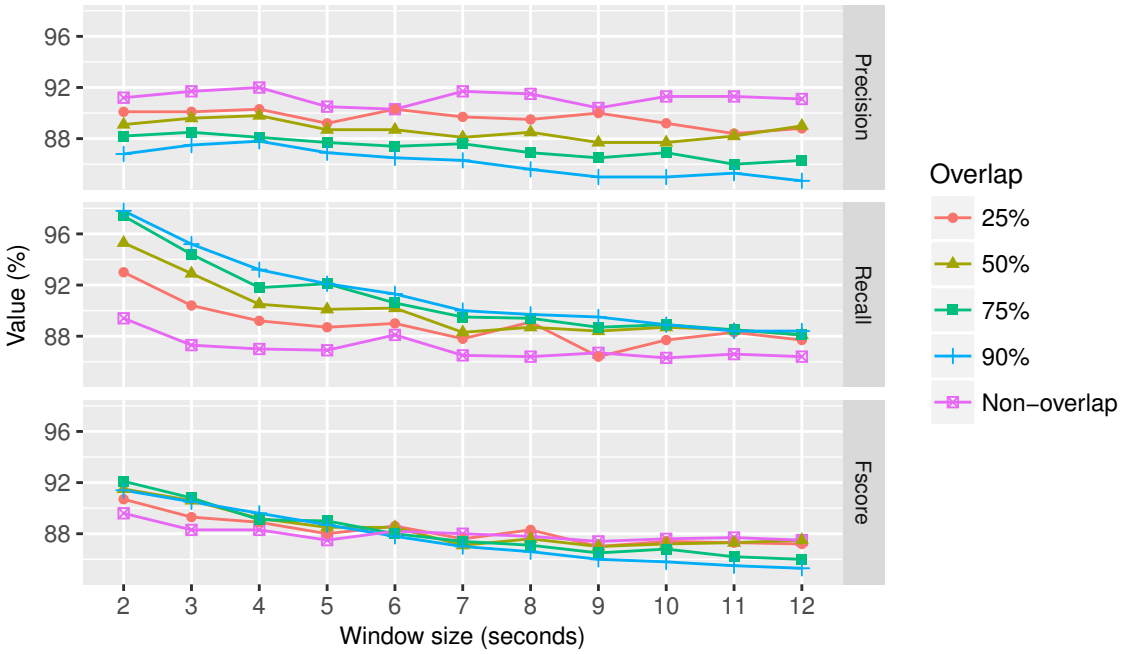


Figure 4.15: FOSW+LR precision, recall, and F-score (%) (Cogent dataset)

degrade the approach's performance, although it is a relatively high improvement when the window is 6 seconds. By looking at the F-score, the overall classifier performance tends to degrade when the window size is increased to 7 seconds, and remains stable afterwards. Figure 4.16 shows the F-scores of FNSW+LR on different window sizes and different C values. The classifier achieves the best result when the size of FNSW is 2 seconds and $C = 10^9$, with a 91.2±11.4% precision, an 89.4±16.2% recall, and an 89.6±12.7% F-score.

Table 4.6 shows the classifier's performance (in terms of precision, recall, and F-score) when FOSW and LR are used, for different overlap sizes. Increasing the overlap size tends to reduce the precision and increase the recall. In terms of F-score, increasing the overlap size tends to improve the classifier's performance. In terms of the window overlap size, the classifier is able to achieve the overall optimal F-score when the data overlap is 50%. Based on this overlap size, the classifier can achieve the best result when the window size is 2 seconds and $C = 10^9$ (see Figure 4.17), with an 89.1±10.9% precision, a 95.3±10.7% recall, and a 91.5±8.7% F-score. Although using 75% of window overlap and a 2-second window can give a better

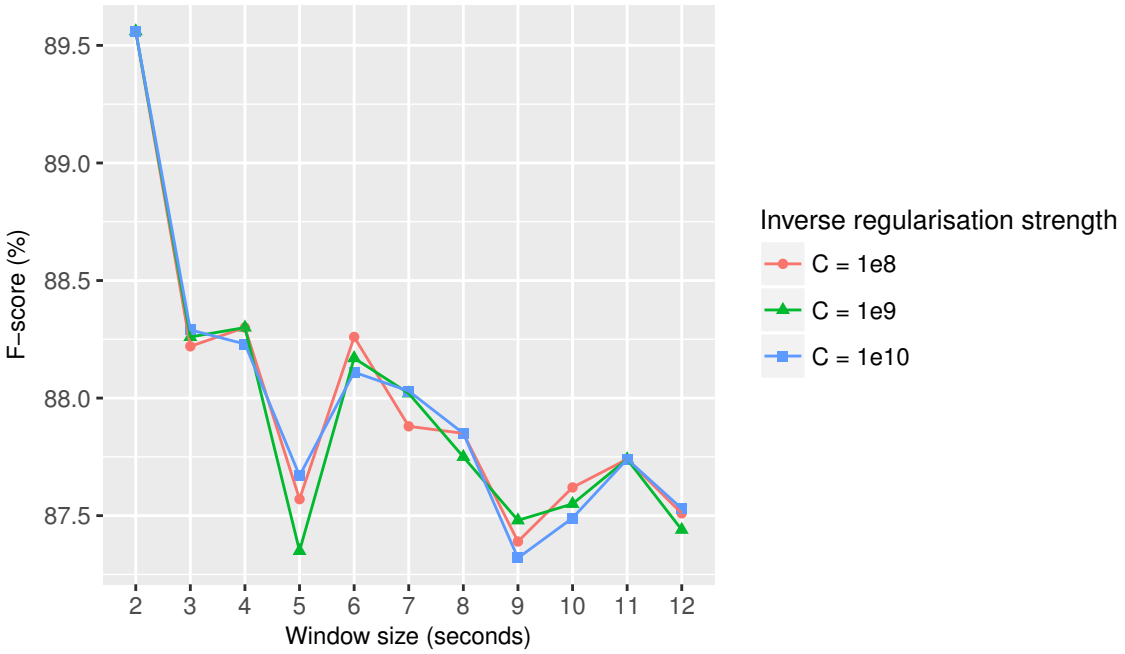


Figure 4.16: FNSW+LR F-score (%) for different C (Cogent dataset)

F-score than using 50% of window overlap and a 2-second window, the difference is not significant (p-value = 0.05).

4.4.3.2 Sliding window + LR performance on the SisFall dataset

Figure 4.18 shows the precision, recall, and F-score of FNSW and FOSW with an LR-based classifier, while Figure 4.19 shows the F-score of FNSW with an LR-based classifier using different values of C. The FNSW+LR-based classifier is able

Table 4.6: FOSW+LR overall performance (average and standard deviation) based on the overlap size using the Cogent dataset

Overlap	**Metric (%)**		
size (%)	**Precision**	**Recall**	**F-score**
25	89.6±11.6	88.8±17.1	88.2±13.1
50	88.6±12.2	90±16.3	88.4±12.8
75	87.3±13.0	90.9±15.8	88.1±12.7
90	86.1±13.8	91.3±15.4	87.7±12.8

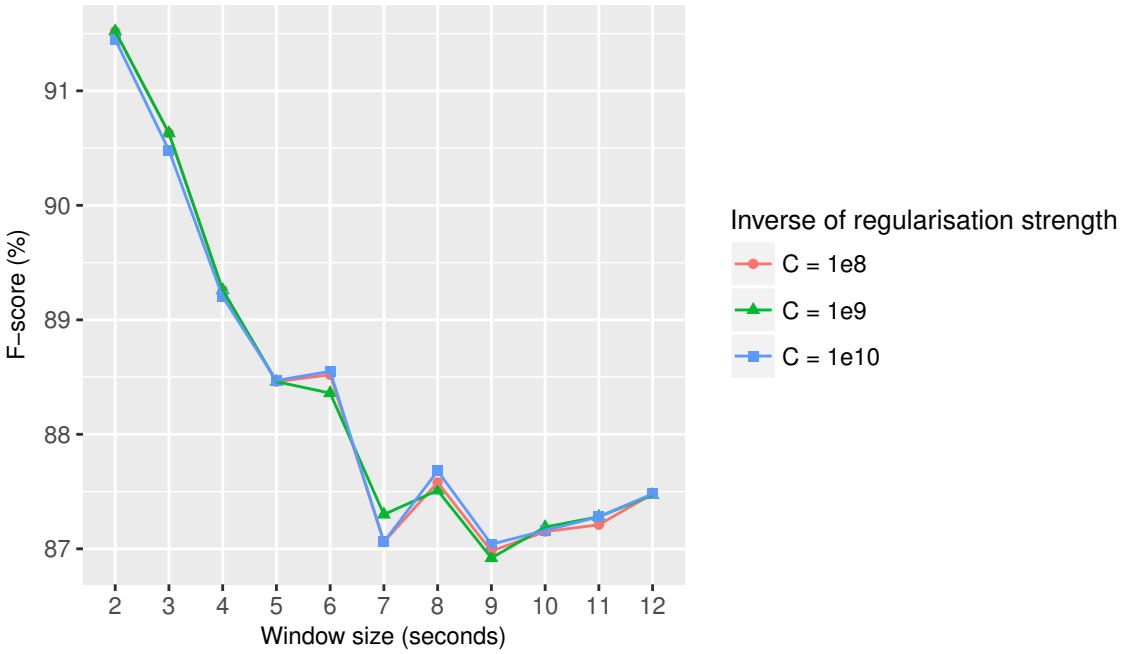


Figure 4.17: FOSW+LR F-score (%) for different C when overlap size is 50% (Cogent dataset)

to achieve better precision and F-score by increasing the window size, while the value of recall remains almost stagnant regardless of the window size. Regarding the value of C, Figure 4.19 shows that the results are similar for three different values of C. The classifier is able to achieve the best result by using a 15-second FNSW with an 87.7±1.1% precision, a 100±0% recall, and a 93.5±0.6% F-score.

Table 4.7 shows that increasing the window overlap for the FOSW+LR-based classifier does not necessarily increase the classifier's performance in terms of precision and F-score. The classifier achieves a lower precision when a bigger window overlap is used. This means that the number of false alarms increases when the window size increases. Figure 4.20 shows the F-score values of FOSW+LR with various inverse-regularisation-strength values when the window overlap is 25%. The classifier can get the best result when the window overlap is 25% and the window size is 15 seconds regardless of C, with an 84.1±1.4% precision, a 100±0% recall, and a 91.4±0.8% F-score.

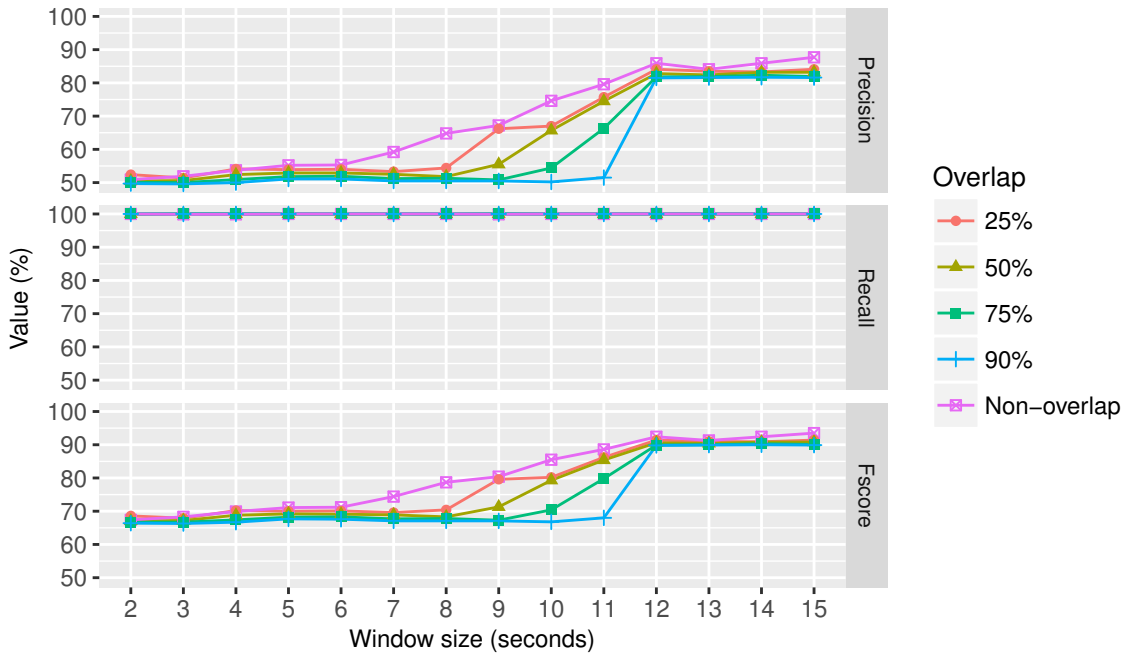


Figure 4.18: FOSW+LR precision, recall, and F-score (%) for different overlap sizes (SisFall dataset)

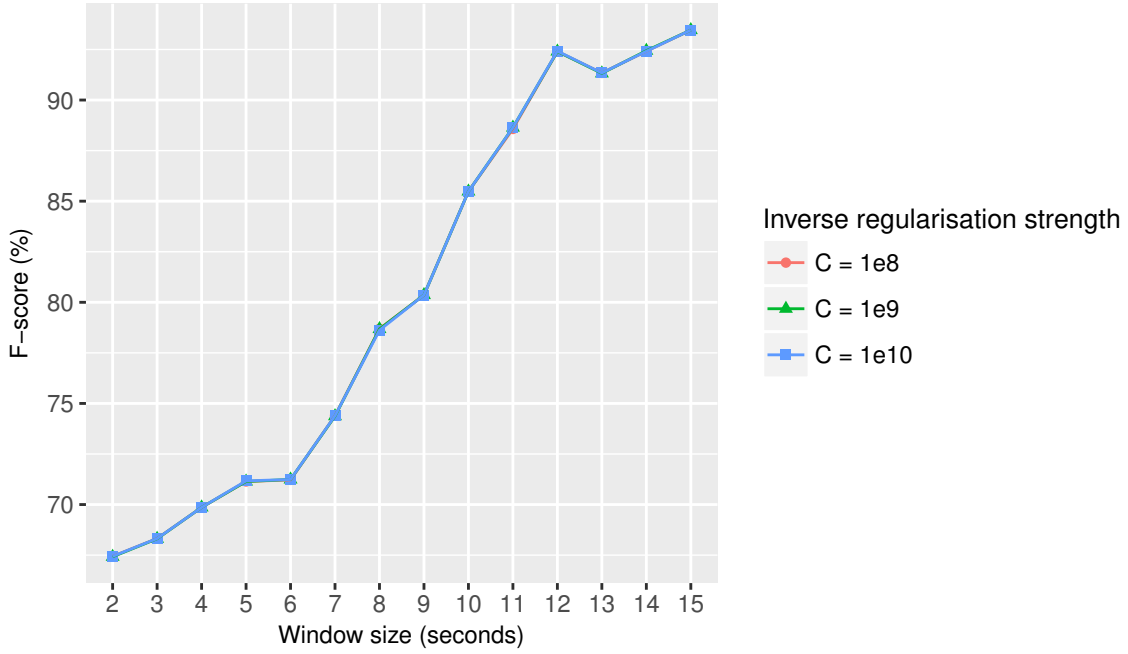


Figure 4.19: FNSW+LR F-score (%) for different C values (SisFall dataset)

Table 4.7: FOSW+LR overall performance (average and standard deviation) based on the overlap size using the SisFall dataset

Overlap	Metric (%)		
size (%)	Precision	Recall	F-score
25	65.5±13.4	100±0.1	78.4±9.6
50	63.6±13.8	100±0.0	76.9±10.0
75	61.2±13.8	100±0.0	75.1±10.0
90	59.4±14.1	100±0.0	73.6±10.3

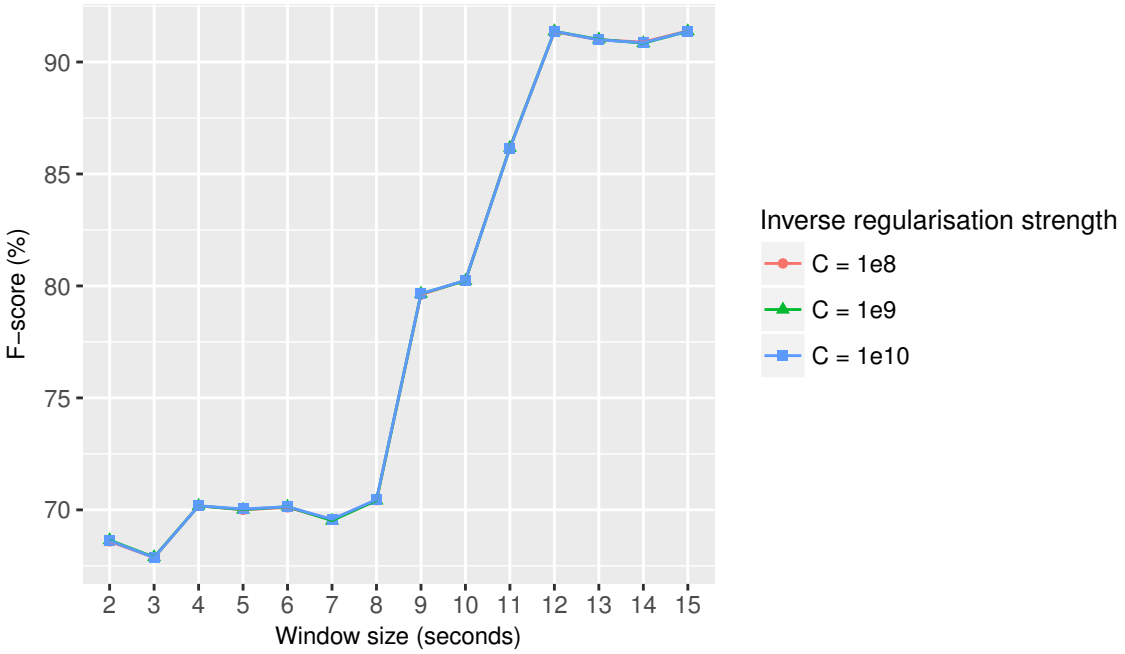


Figure 4.20: FOSW+LR F-score (%) for different C when overlap size is 25% (SisFall dataset)

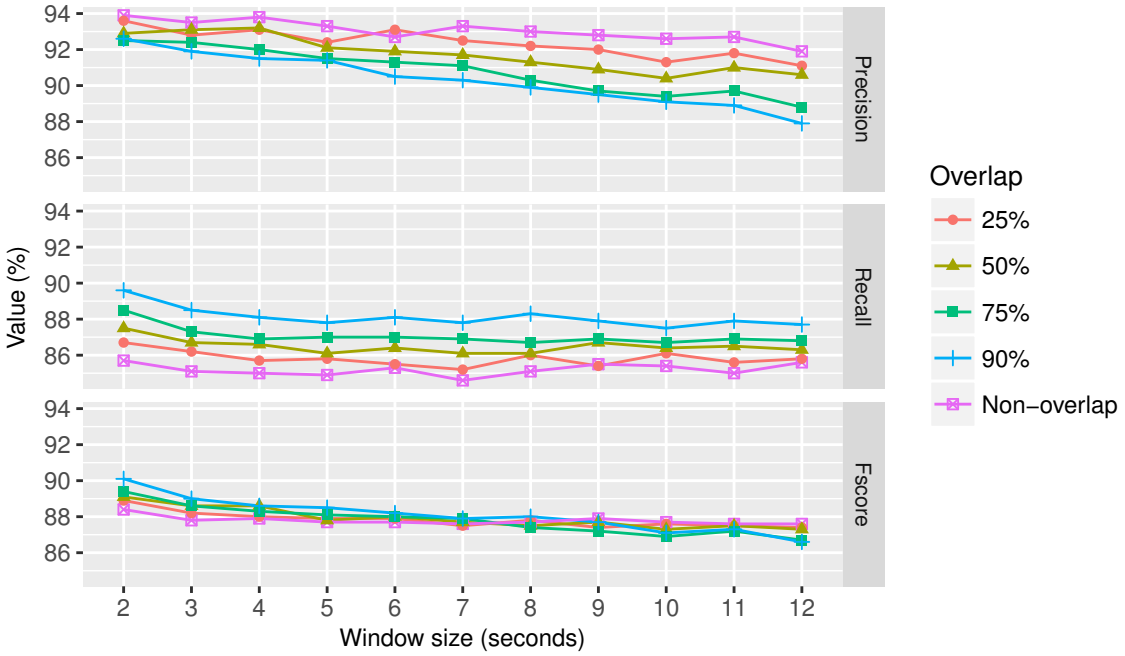


Figure 4.21: FOSW+SVM F-scores (%) for different overlap sizes (Cogent dataset)

4.4.4 Support vector machine (SVM) based fall-detection approach

4.4.4.1 Sliding window + SVM performance on the Cogent dataset

The impact of the window size on the precision, recall, and F-score of the FNSW+SVM- and FOSW+SVM-based classifier is shown in Figure 4.21. In terms of precision, the classifier's performance tends to degrade when the window size increases. The recall tends to remain stable when the window size is increased. Overall, the F-score decreases when the window size increases from 2 seconds to 3 seconds and tends to remain stable afterwards. The classifier can achieve the best result when the window size is 2 seconds. Figure 4.22 shows the FNSW+SVM-based classifier F-scores for different window sizes and different kernel types. Based on these scores, the classifier is able to get the best performance when the window size is 2 seconds and the kernel type is RBF, with a 93.1±10.7% precision, an 87.4±18% recall, an 89.2±13.8% F-score.

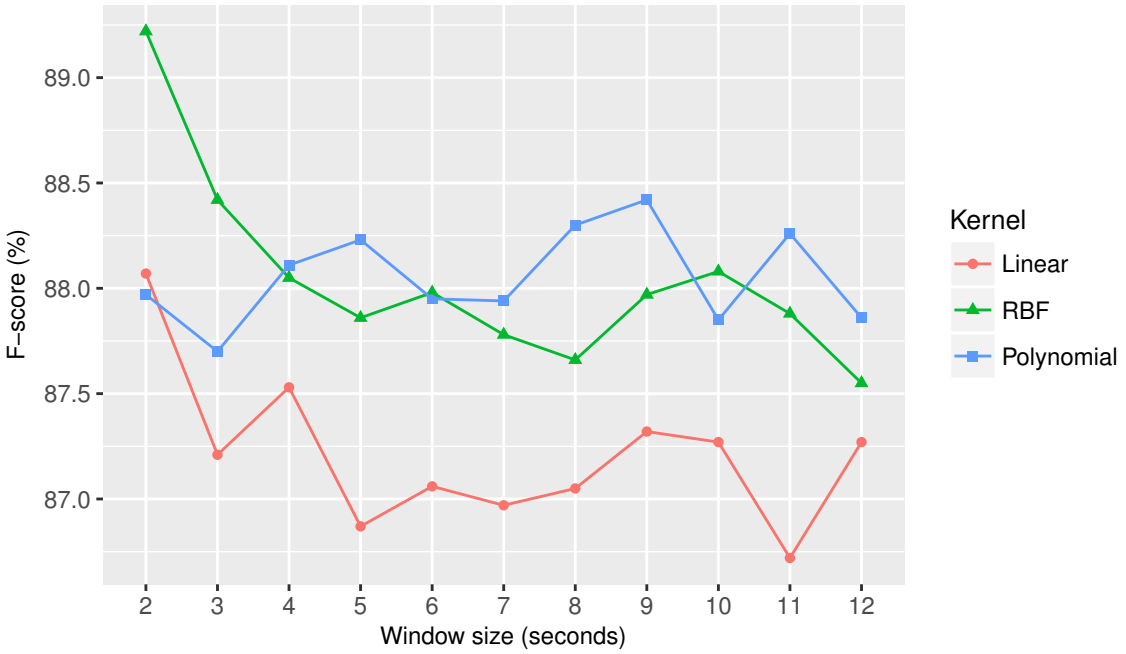


Figure 4.22: FNSW+SVM F-scores (%) for different kernels (Cogent dataset)

Table 4.8 provides precision, recall, and F-score values for the FOSW+SVM-based approach. In terms of precision, the classifier performance decreases when the overlap size increases. On the other hand, the recall increases when the overlap size increases. Overall, the classifier performance, in terms of F-score, remains stable when the overlap size increases. The classifier is able to reach the optimal result when the data overlap is 90%. Figure 4.23 shows the F-scores of the FOSW+SVM-based classifier for different kernel types. The classifier can achieve the best performance when the window size is 2 seconds and the kernel type is RBF, with a 92.1±10.1% precision, a 93.6±12.6% recall, and a 92.3±9.7% F-score on average.

4.4.4.2 Sliding window + SVM performance on the SisFall dataset

Figure 4.24 shows the precision, recall, and F-score values of different sizes of FNSW and FOSW with an SVM-based classifier, while Figure 4.25 shows the F-score values of the FNSW+SVM-based classifier with different kernels. By increasing the window size, the precision and F-score of the FNSW+SVM-based classifier increases, while the recall remains stable. This classifier can reach the optimum performance when

Table 4.8: FOSW+SVM performance (average and standard deviation) analysis based on the overlap size using the Cogent dataset

Overlap	**Metric (%)**		
size (%)	**Precision**	**Recall**	**F-score**
25	92.3±10.6	85.8±18.7	87.8±14.4
50	91.7±11.4	86.5±18.4	87.9±14.2
75	90.8±12.4	87.0±18.0	87.8±14.3
90	90.3±88.1	88.1±17.6	88.1±14.3

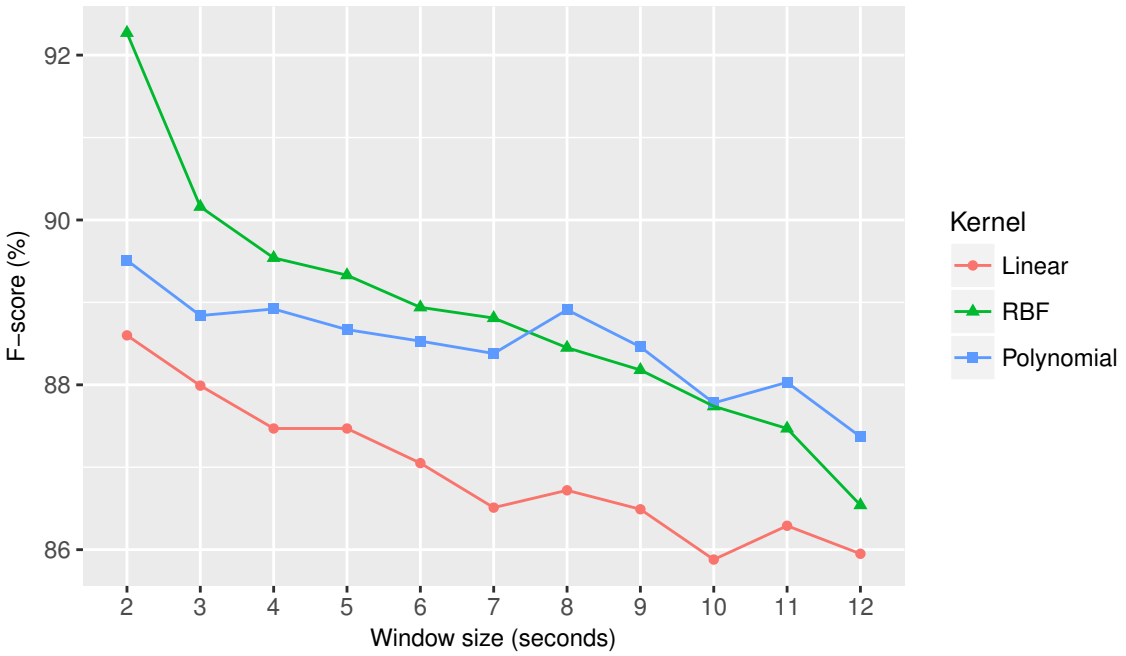


Figure 4.23: FOSW+SVM F-scores (%) for different kernels when overlap size is 90% (Cogent dataset)

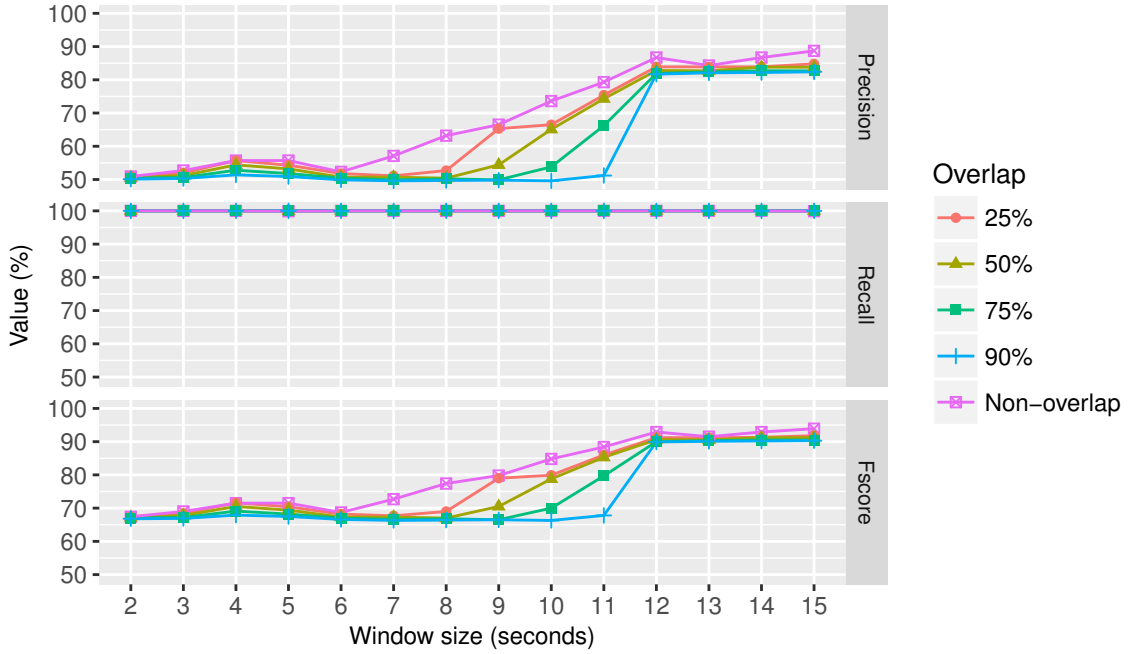


Figure 4.24: FOSW+SVM F-scores (%) for different overlap sizes (SisFall dataset)

15 seconds of window size and a polynomial kernel are used. With this setup, the classifier can achieve a 90.1±1.8% precision, a 99.9±0.4% recall, and a 94.7±1.0% F-score.

Table 4.9 shows the precision, recall, and F-score of FOSW with an SVM-based classifier. Similarly to the previous results, these results show that increasing the window overlap does not necessarily improve the classifier's performance. The best window overlap to use is 25%. With regard to the SVM kernel, the polynomial kernel can give the best performance (see Figure 4.26) with an 86.5±2.6% precision, a 100±0% recall, and a 92.8±1.5% F-score on average, when the window size is 15 seconds.

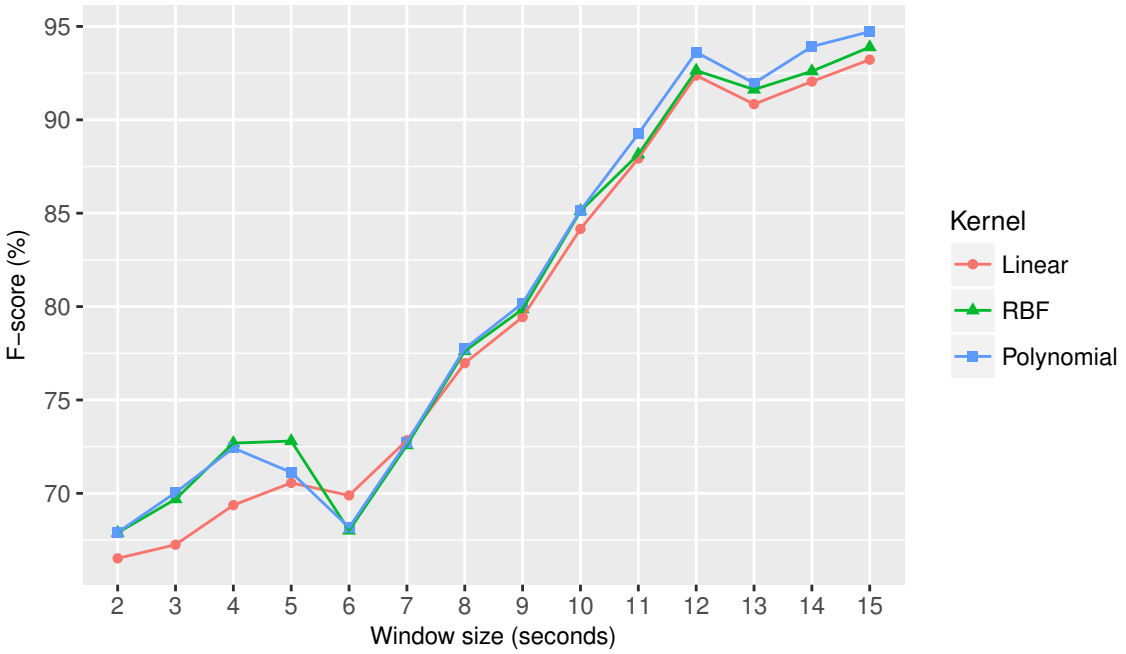


Figure 4.25: FNSW+SVM F-score (%) for different kernels (SisFall dataset)

Table 4.9: FOSW+SVM overall performance (average and standard deviation) based on the overlap size using the SisFall dataset

Overlap	**Metric (%)**		
size (%)	**Precision**	**Recall**	**F-score**
25	65.2±13.9	100±0.1	78.1±10.0
50	63.4±14.2	100±0.0	76.7±10.2
75	61.1±14.1	100±0.0	75±10.3
90	59.1±14.3	100±0.0	73.5±10.5

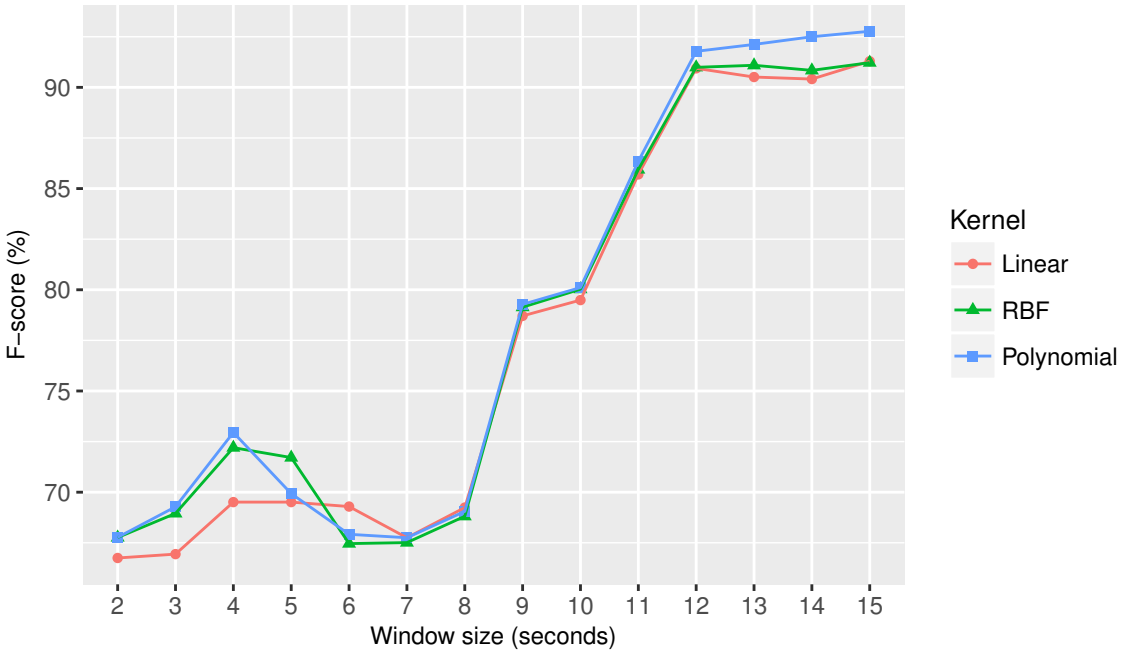


Figure 4.26: FOSW+SVM F-scores (%) for different kernels when overlap size is 25% (SisFall dataset)

4.5 Analysis, discussion, and limitations

4.5.1 Threshold-, FNSW-, and FOSW-based approaches' performance

For the machine-learning-based fall-detection approaches, the following tendencies are retrieved from the results of both the FNSW- and FOSW-based machine-learning approaches tested on the Cogent dataset:

- Precision tends to increase when the window size increases for the FNSW-based machine-learning approach when it uses CART or k-NN. However, the precision tends to decrease when the window size increases if the FNSW-based machine-learning approach uses LR or SVM. For the FOSW-based machine-learning approaches, the precision decreases when the overlap size increases for most cases.

- Recall tends to decrease when the window size increases for the FNSW-based

machine-learning approach, except when it uses an SVM. For the FOSW-based machine-learning approaches, the recall increases when the overlap size increases, in all cases.

- In terms of the F-score, increasing window size is only effective to increase the classifier's performance when the FNSW-based machine-learning approach uses CART or k-NN. For the FOSW-based machine-learning approaches, the F-score tends to decrease when the overlap size increases and CART, k-NN, or LR is used as the machine-learning algorithm. On the other hand, when the FOSW-based machine-learning approach uses an SVM, the F-score tends to increase when the window overlap size increases.

The following tendencies are retrieved from the results of both FNSW- and FOSW-based machine-learning approaches tested on the SisFall dataset:

- Precision and F-score tend to increase when the window size increases for the FNSW-based machine-learning approach, regardless of the machine-learning algorithm. However, the precision tends to decrease when the window overlap increases for the FOSW-based machine-learning approaches in all cases.
- Recall tends to be steady for both FNSW- and FOSW-based approaches.

From the information above, it can be seen that increasing either the window size or the overlap size does not necessarily improve the classifier's performance (in terms of precision, recall, and F-score) for the Cogent dataset. Figure 4.27 shows a summary of the F-score for each machine-learning algorithm tested on the Cogent dataset. For the SisFall dataset, increasing the window size can increase the precision and F-score. This is because the lengths of falls are uniform (15 seconds). Therefore, using a larger window size can increase both the precision and the F-score. In reality, the lengths of falls are unpredictable. This makes the results from the SisFall dataset not able to represent the performance of the classifier in a real-world case. Thus,

the results from the Cogent dataset are more relevant to represent the classifier's performance in real-world cases. For the Cogent dataset, the best window size for both FNSW and FOSW that can give the optimum F-score is 2 seconds. This size is also recommended by Banos *et al.* [13], where they recommend a window size that range from 0.25 to 3.25 seconds for recognising an energetic activity (including falls).

Another finding from this study suggests that increasing the overlap size can cause the number of false alarms to increase for both datasets. Figure 4.28 shows the feature distributions when the FOSW-based machine-learning approach uses 25% and 90% window overlap size. These figures show that increasing the overlap of the window can increase the number of data overlaps between fall and non-fall data. This can cause the classifier to mistakenly learn that the non-fall samples are falls during the training, and can make the classifier mistakenly classify non-falls as falls during the testing. Figure 4.28 also shows that choosing a correct window size is important. Ideally, fall activities produce higher acceleration-magnitude-mean values than some other non-fall activities such as lying on the bed or sitting on the chair (Sub-figures 4.28a and 4.28b). However, an anomaly arises when the SisFall dataset is used (shown in Sub-figures 4.28c and 4.28d). This is caused by a gap between the window size and the length of the fall (a 3-second FOSW is used to generate these figures, while the length of the fall is 15 seconds). This gap allows a segment to not have any peak at all, although that segment is annotated as a fall. This condition happens during the post-impact stage, where the data are annotated as a fall but the body of the subject produces low acceleration values. Segments 3, 4, 5 in Figure 2.4 are examples of this condition.

In response to **RQ1**- increasing the window size of the FNSW does not necessarily improve the classifier's performance (in terms of precision, recall, and F-score), unless the lengths of all falls are fixed and uniform. Increasing the overlap size of the FOSW can decrease the precision in general.

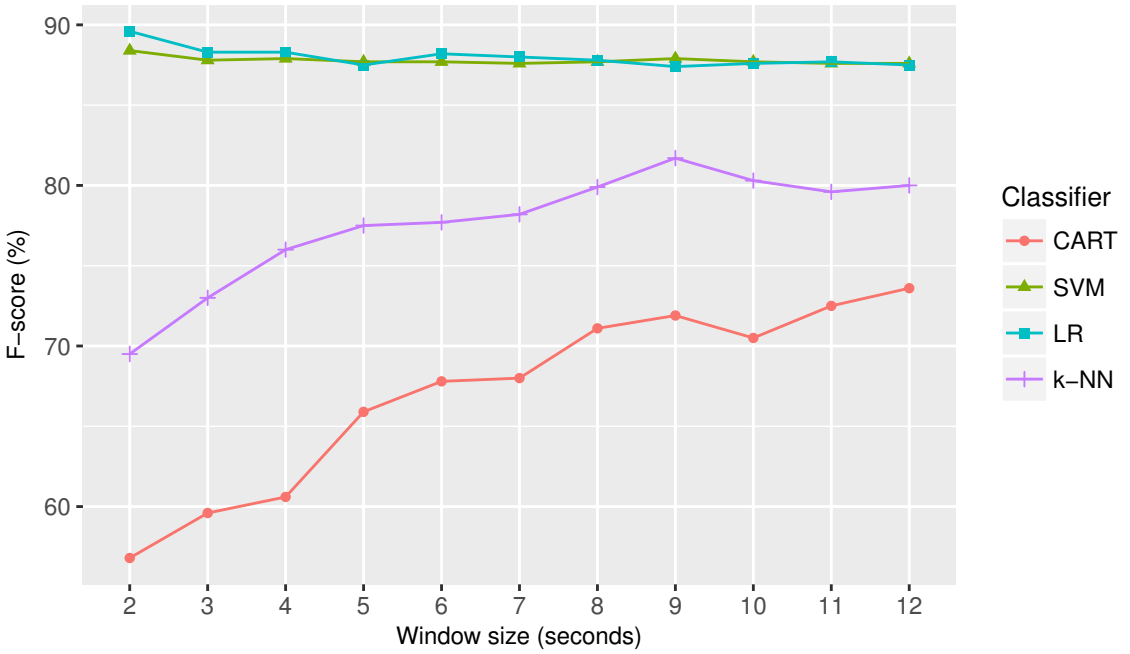
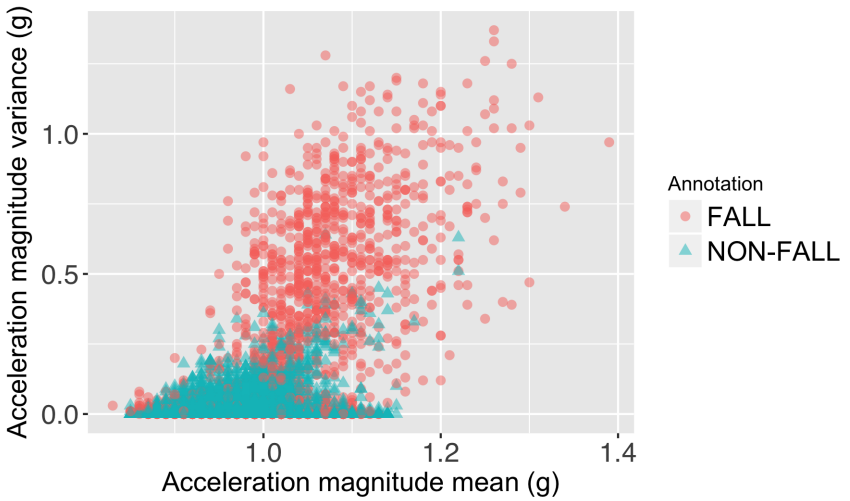


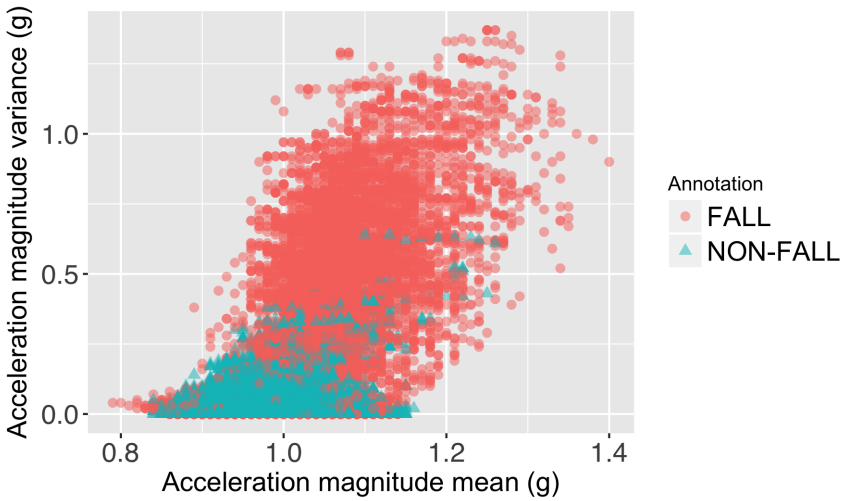
Figure 4.27: A summary of F-scores of FNSW for each machine-learning algorithm.

In general, from the experiments above, both FNSW- and FOSW-based machine-learning approaches are unable to give a balanced performance between precision and recall. Having a balanced precision and recall is important, as detecting falls accurately and reducing the number of false alarms is important. Having a high number of false alarms can cause the users to reject the system [115]. When the number of false negatives increases, then the number of recalls decreases. A reduction in the recall can cause the system to mis-detect falls, and it can cause the patient to be left helpless.

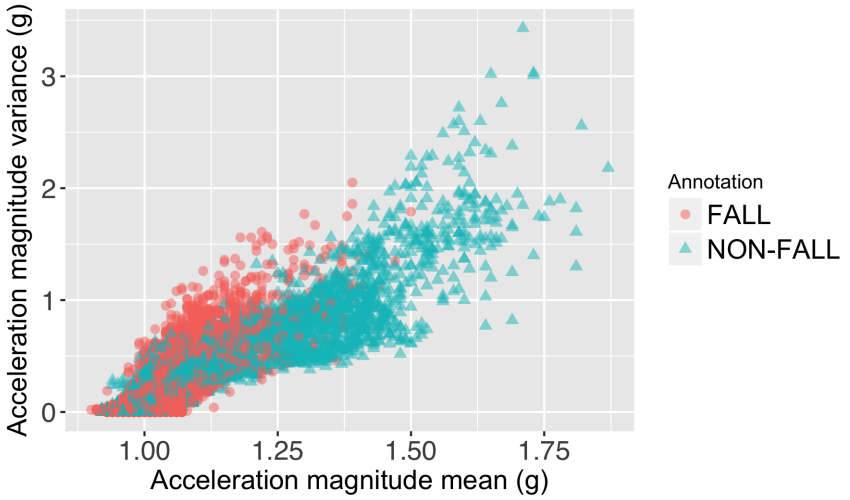
Table 4.10 shows a comparison between the IMPACT+POSTURE and sliding-window approaches' F-scores using two publicly accessible datasets (the Cogent and SisFall datasets). This comparison uses the best results from the sliding-window-based approaches. The FNSW-based machine-learning approaches take results from the LR-based classifier (with $C = 10^9$) for the Cogent dataset, while results from the k-NN-based (with $k = 2$) classifier are taken for the SisFall dataset. The FOSW-based machine-learning approach takes results from the SVM-based classifier (with



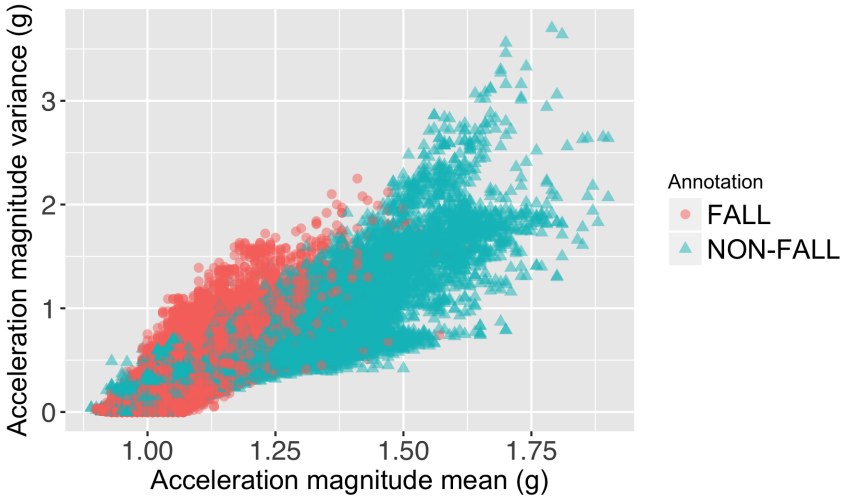
(a) Feature values distribution of 25%-FOSW on Cogent dataset.



(b) Feature values distribution of 90%-FOSW on Cogent dataset.



(c) Feature values distribution of 25%-FOSW on SisFall dataset.



(d) Feature values distribution of 90%-FOSW on SisFall dataset.

Figure 4.28: Data overlaps caused by increasing the window overlap on FOSW

an RBF kernel) for the Cogent dataset, while results from the k-NN-based classifier (with $k = 2$) are taken for the SisFall dataset.

For the Cogent dataset, by looking at the F-score, the FNSW+LR-based approach is able to outperform POSTURE+IMPACT, although the difference is not significant (with p-value = 0.8). Compared to IMPACT+POSTURE, the FOSW+SVM-based approach is able to achieve a slightly better performance, with p-value = 0.3. This means that the machine-learning-based approaches are able to outperform IMPACT+POSTURE on the Cogent dataset, although the difference is not significant. For the SisFall dataset, the FNSW+k-NN-based approach is able to significantly outperform IMPACT+POSTURE (with p-value = 4.3×10^{-5}). The FOSW+k-NN-based approach is also able to achieve a significantly better F-score than IMPACT+POSTURE (with p-value = 4.3×10^{-5}) tested on the SisFall dataset.

The results in Table 4.10 show that using either FNSW- or FOSW-based approaches can give a better overall performance (F-score) than POSTURE+IMPACT. In response to **RQ**2- yes, the sliding-window-based machine-learning approach outperforms (in terms of F-score) the threshold-based approach on publicly accessible datasets, although the difference is not significant when the Cogent dataset is used, regardless of the sliding-window technique. This result is also supported by the results provided in Aziz et al.'s [8] study, where their study compares FNSW-based machine-learning approaches and five different threshold-based approaches. Another study about the optimum window size for the machine-learning-based fall-detection approaches was conducted by Aziz *et al.* [9]. However, this study focuses only on finding the optimum window size for the pre-impact stage of the fall.

4.5.2 Limitations

This study found that there is an annotation problem, especially for the SisFall dataset, for which the fall-events data are not precisely annotated. Figure 4.29 shows the annotation problem of the SisFall dataset. This problem becomes a critical issue

Table 4.10: F-scores comparison between IMPACT+POSTURE and sliding window-based approaches

Approach	F-score (%)	
	Cogent	**SisFall**
IMPACT+POSTURE	88.6±9.2	70.2±1.4
FNSW-based	89.6±12.7	96.5±1.4
FOSW-based	92.3±9.7	95.3±1.4

when the simulation is implemented in a real-time style, while it is not a big issue for an off-line style simulation. This is because the data stream is pre-segmented based on its annotation in the off-line style simulation. A re-annotation process cannot be done in this thesis because the SisFall dataset does not provide videos of all subjects, where these videos are the ground truth. In fact, annotating accelerometer data is more difficult than annotating data from a camera [30]. The Cogent dataset has a better annotation than the SisFall dataset. Thus, the results from the Cogent dataset can represent the performance of the classifier in real-world cases better than the results from the SisFall dataset. A lesson learnt from this problem is that annotating the data stream precisely is very important.

Since the features can affect the performance of the classifier, the results that are provided in this chapter may be different with studies that use different types of features. The type of machine-learning algorithm used to train the classifier can also produce different results. Since the main focus of this chapter is to examine the impact of the window size and overlap, improving the classifier performance by tuning its parameters is out of the scope of this study and it is considered for future work.

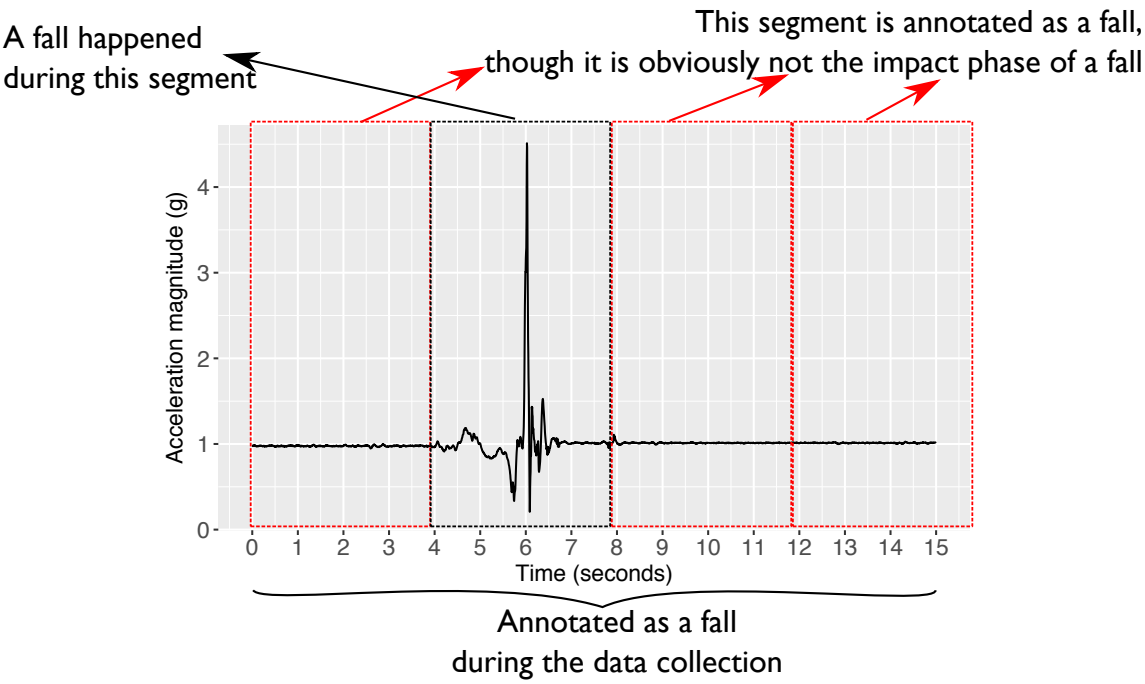


Figure 4.29: The annotation issue on the SisFall dataset

4.6 Chapter summary

This chapter provides an analysis of two types of fall-detection approach: threshold- and sliding-window-based machine learning. To extract features from a data sequence, a machine-learning-based fall-detection approach uses either a fixed-size non-overlapping sliding window (FNSW) or a fixed-size overlapping sliding window (FOSW). For the FNSW-based fall-detection approach tested on the Cogent dataset, the best result is achieved when the window size is 2 seconds and the machine learning used is LR (with $C = 10^9$), with an 89.6% F-score. For the SisFall dataset, the best result is achieved when the window size is 15 seconds and the machine learning used is k-NN (with $k = 2$), with a 96.5% F-score. For the SisFall dataset, having a larger window can increase the precision, recall, and F-score. However, this result is biased, as the length of an activity in real cases is unpredictable. Thus, using the SisFall dataset to define the optimum window size is not appropriate.

The FOSW-based machine-learning approach can achieve its best performance on the Cogent dataset when this approach uses the window size of 2 seconds, the

overlap size of 90%, and SVM (with an RBF kernel) to train the classifier. For the SisFall dataset, this approach is able to achieve an up to 95.3% F-score when the length of the window is 15 seconds, the size of the window overlap is 25%, and a k-NN-based classifier ($k = 2$) is used to build the classifier.

As a summary, this thesis retrieves two important findings related to the sliding-window technique from the analysis of the machine-learning-based approaches:

- Using a larger FNSW does not necessarily increase the precision, recall, and F-score of the classifier, unless the lengths of the falls are fixed and uniform.
- Using an FOSW with a larger overlap size can cause the precision and the F-score to decrease in most cases.

Those findings above can cause the current machine-learning-based approaches to have a gap between precision and recall values, while having balanced precision and recall is important for real-world implementation.

The main drawback of the threshold-based approach is the difficulty in defining its thresholds. Manually defining thresholds can cause the number of false alarms or undetected falls to be high, as there are overlaps between fall and non-fall data. The experiments show that POSTURE+IMPACT (one of the threshold-based approaches) can achieve up to 88.6% F-score. In general, using the sliding-window-based machine-learning approach can give a better F-score than using the threshold-based approach. However, the difference is not significant when the Cogent dataset is used. Thus, the next chapter proposes a novel machine-learning-based approach that can significantly improve the precision, the recall, and the F-score of the machine-learning based approach for fall detection. This novel approach is also able to reduce the computational cost of the system, and can give an advantage when this approach is implemented on a real wearable device.

Chapter 5

Event-triggered machine-learning approach (EvenT-ML)

5.1 Introduction

The previous chapter analyses both threshold- and machine-learning-based approaches using the Cogent and SisFall datasets. Since the main issue of the machine-learning based approaches is the way data are segmented and features are extracted, this chapter provides a novel approach called Event-triggered machine-learning approach (EvenT-ML) that proposes a fall-stage-based segmentation approach. This approach can improve the detection rate of the classifier (in terms of precision, recall, and F-score) and reduce the computational cost (in terms of running time).

To improve the classifier's detection rate (in terms of precision, recall, and F-score), Ojetola [118] and Putra *et al.* [127] have shown that features should be extracted based on fall stages (pre-impact, impact, and post-impact), as every stage has its own characteristic. During the pre-impact stage, acceleration drops below $1g$ as the subject experiences free fall after losing their balance. The impact stage usually yields one or more high peaks as a result of an impact between the subject's

body and the ground. The post-impact stage is usually characterised by inactivity, corresponding to reduced variation in the accelerometer reading.

Although Ojetola's technique is able to improve the precision and recall of the machine-learning-based approach, this technique has a high computational cost [127]. It uses an FOSW with an overlap of $N-1$, where N is the total number of samples in the window. A study from Bersch *et al.* [19] found that using an overlapping sliding window with a high offset can increase the computational cost, as the system needs to do the feature extraction and classification processes more often. Kau *et al.* [92] show that calculating complex features can increase the system's computational cost. Having a high-computational-cost system on a wearable device can drain the battery quickly. Beside the computational cost, the previous chapter shows that using a sliding window with a relatively large window overlap is not effective, as it can increase the data overlaps between fall and non-fall activities.

To reduce the computational cost of Ojetola's technique, Putra *et al.* [127] developed a cascade-classifier approach (CCA). The main idea of the CCA is to prevent the system from doing the feature extraction all the time, by checking the state of the subject and performing the feature extraction only when the system detects an energetic event. The system determines an energetic activity by checking whether the data sequence has any peaks higher than $1.6g$ during 2 seconds of window. By using the peak-detection mechanism, CCA is not only able to reduce the computational cost but also can increase the classifier performance compared to Ojetola's technique. Although CCA is able to improve the performance of the system in terms of computational cost and detection rate, the identification of the temporal position of stages is not simple.

Segmenting a data stream based on fall stages is a challenging task as it is difficult to determine the beginning and the end of each stage in a segment. use high acceleration peaks to estimate the impact stage. In fact, during the fall, the body of the subject produces several high peaks of acceleration which can confuse

the segmentation process. Furthermore, Jamsa *et al.* [82] show that peaks can also appear during the pre-impact stage as a result of protective actions. The presence of multiple peaks (called *the multi-peak issue*, see subsection 2.5.4 for a detailed explanation of the multi-peak issue) makes the estimation of the impact stage even harder, because it can cause a misalignment of this stage. Although this multi-peak issue is critical for the feature-extraction process, it has not been considered in either of Ojetola and Putra *et al.*'s studies.

To reduce the computational cost and resolve the multi-peak issue, this thesis developed a new technique called the event-triggered machine-learning approach (EvenT-ML). This technique consists of:

- The use of a finite state machine to segment a data sequence based on three fall stages: pre-impact, impact, and post-impact, where the feature extraction process uses these stages as its basis. This concept is relatively new in machine-learning-based fall-detection studies, since only the studies from Ojetola [118] and Putra *et al.* [127] from Table 2.2 use this concept. Although the Ojetola and Putra *et al.* approaches also use fall stages as a basis for feature extraction, their approaches do not provide a mechanism to correctly estimate the beginning and the end of each stage. In fact, the multi-peak issue can cause a misalignment of the impact stage.

- A mechanism to resolve the ambiguity caused by multiple peaks so that the alignment of each stage of the fall can be estimated. Although Abbate *et al.*'s [2] study provides a mechanism to avoid the multi-peak issue, their approach does not provide a mechanism to estimate the beginning and the end of the fall stages and extract features from the stages. This chapter shows that extracting features based on fall stages can significantly improve the F-score of the classifier.

An event-based-type technique is common in the machine-learning-based fall-detection

Table 5.1: Comparison of EvenT-ML with literatures

Approach	Fall-stage-based feature extraction	Multi-peak detection
Gjoreski *et al.* [64]	✓ (impact and post-impact)	✗
Abbate *et al.* [2]	✗	✓
Ojetola [118]	✓ (pre-impact, impact, and post-impact)	✗
Putra *et al.* [127]	✓(pre-impact, impact, and post-impact)	✗
Kau *et al.* [92]	✗	✗
EvenT-ML	✓	✓

approaches (this type of approach is similar to the threshold-machine-learning-based approach in Subsection 2.3.3), Table 5.1 shows the comparison between EvenT-ML with the literatures from Subsection 2.3.3.

This chapter includes:

1. Event-triggered machine-learning approach (EvenT-ML).

2. Five comparative evaluations. The first evaluation aims to compare FNSW- and FOSW-based machine-learning approaches with EvenT-ML. To see the improvement of EvenT-ML that is made by solving the multi-peak issue, the second evaluation compares CCA with EvenT-ML. Then, the third evaluation compares the EvenT-ML and IMPACT+POSTURE approaches. The fourth evaluation evaluates the performance, in terms of precision, recall, and F-score, of EvenT-ML with three different sensor placements. This fourth evaluation only applies to the Cogent dataset, as the SisFall dataset only has one sensor placement. The last evaluation provides an evaluation of the precision, recall, and F-score of EvenT-ML on real-fall data from older patients by using the FARSEEING dataset.

The structure of this chapter is as follows: Section 5.2 provides the method of this chapter. Section 5.3 explains Event-ML, while Section 5.4 provides results and analysis of the experiments. Section 5.5 provides a discussion and the limitations of this chapter, and Section 5.6 concludes this chapter.

5.2 Method

This chapter reports experiments to evaluate EvenT-ML, where the Cogent, SisFall, and FARSEEING datasets were used in this evaluation. See Chapter 3 for information regarding those datasets. The evaluation consists of four parts:

- A comparison between EvenT-ML, FNSW-, and FOSW-based machine-learning approaches on both the Cogent and SisFall datasets;
- A comparison between EvenT-ML and existing fall-detection approaches (CCA-based machine-learning approaches from Putra *et al.* [127] and IMPACT+POSTURE from Kangas *et al.* [86]) on both Cogent and SisFall datasets. The comparison between EvenT-ML with CCA aims to show that solving the multi-peak issue can significantly improve the classifier performance (in terms of F-score), while the comparison between EvenT-ML with IMPACT+POSTURE aims to show that a machine-learning-based approach (regardless of the machine-learning algorithm) can outperform a threshold-based approach only when the data is segmented correctly based on the fall stages;
- EvenT-ML's performance with different sensor placements. To evaluate this performance on several different placements, a subset from the Cogent dataset is used, as not all subjects have waist-sensor data. Table 3.1 shows the body profiles of the subjects in this subset.
- EvenT-ML's performance on real-fall data from older patients. This part involves the FARSEEING dataset.

This chapter uses the same machine-learning algorithms as in the previous chapter: Classification and Regression Tree (CART), Logistic Regression (LR), Support Vector Machine (SVM), and k-Nearest Neighbour (k-NN), for training and testing the classifier. Also, this chapter uses a real-time style simulation, where the sample ap-

pears one by one (see Figure 4.1). To evaluate the performance of EvenT-ML compared to sliding-window-based approaches (Subsection 5.4.1), CCA-based machine-learning approach (Subsection 5.4.2), and IMPACT+POSTURE (Subsection 5.4.3), LOSOCV is used. For evaluating EvenT-ML's performance on the FARSEEING dataset, the following schemas are used:

- Using the Cogent dataset to build the classifier, then using the FARSEEING dataset as the test set (Figure 5.1),
- Using the SisFall dataset to build the classifier, then using the FARSEEING dataset as the test set (Figure 5.1), and
- Using the FARSEEING dataset for both training and testing using the LOSOCV technique (Figure 5.2).

The hold-out method is suitable for measuring the effectiveness of using data from young and healthy subjects to detect falls in older people, by assuming that the data from older people are not available during the training process. Figure 5.2 shows the hold-out technique from this study. This evaluation method uses the data from young and healthy subjects as a training set and uses the data from older patients as a testing set.

This study uses precision, recall, and F-score as the performance metrics. See formulas (2.2), (2.1), and (2.3) to calculate precision, recall, and F-score. To evaluate activities that are often misclassified as falls and falls that are often misclassified as non-falls, this study uses the false positive ratio (FPR) and false negative ratio (FNR):

$$FPR_i = \frac{\mathrm{FP}_i}{p}, \tag{5.1}$$

$$FNR_i = \frac{\mathrm{FN}_i}{q}, \tag{5.2}$$

where FP_i, FN_i, p, q, and i are the number of times that the i^{th} type of activity is

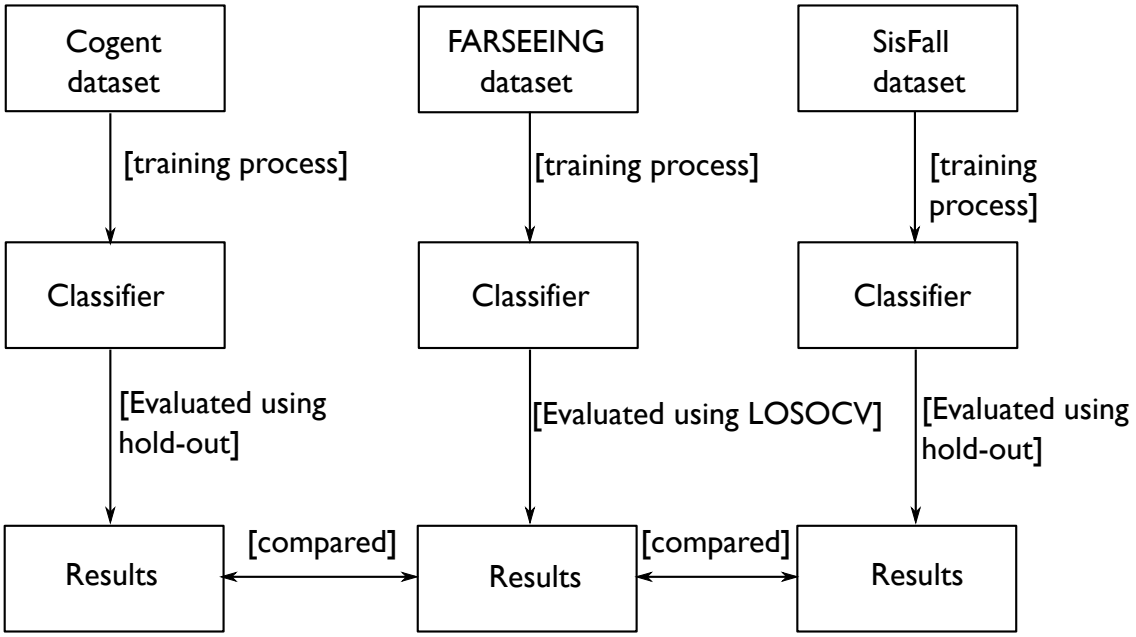


Figure 5.1: A comparison study between the Cogent, SisFall, and FARSEEING datasets

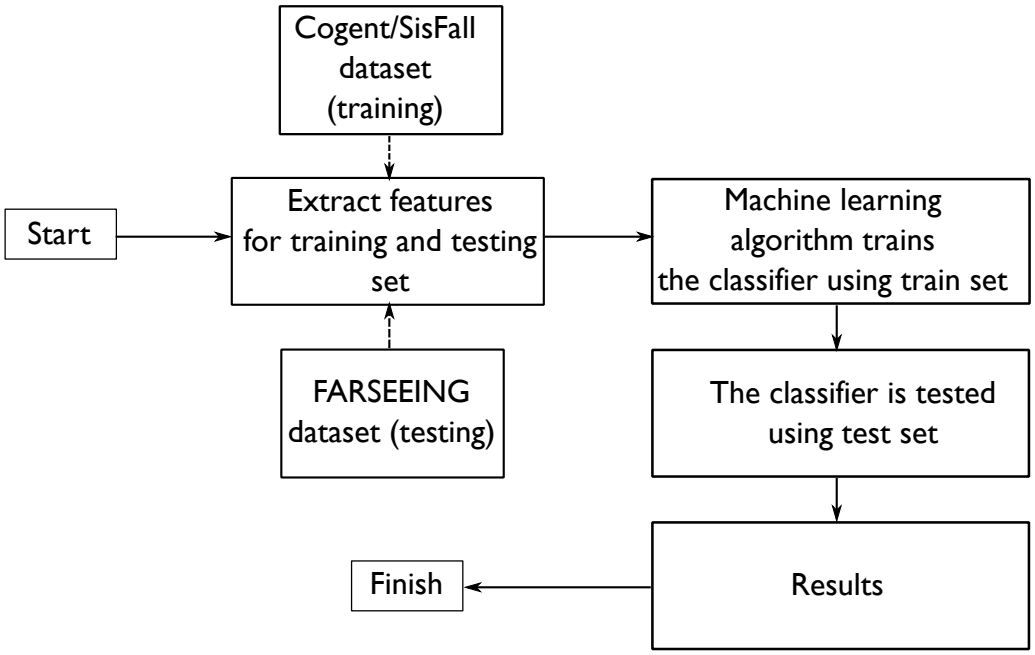


Figure 5.2: Hold-out evaluation technique

misclassified as a fall, the number of times that the i^{th} type of fall is misclassified as a non-fall, the total number of samples of the i^{th} type of activity, the total number of samples of the i^{th} type of fall, and the type of fall or activity. An activity/fall with a higher FPR/FNR means that that particular type of activity/fall is harder to detect.

Beside focusing on improving the accuracy of the classifier, EvenT-ML also aims to reduce the computational cost of the sliding-window-based (FNSW and FOSW) machine-learning approaches. According to Bersch *et al.* [19], the computational cost of the classification system consists of two parts:

$$CC = Stage_1 + Stage_2, \tag{5.3}$$

where $Stage_1$ and $Stage_2$ are the computational costs of the data pre-processing (data segmentation and feature extraction) and the classification, respectively. As this chapter focuses on the segmentation and feature-extraction issues, only $Stage_1$ is considered. The experiment was done offline on a PC with the following specifications:

- Operating system: 64-bit Linux Ubuntu 14.04 LTS,
- Memory: 15.6 GB,
- Processor: Intel Core i7-3770K CPU @ 3.5 GHz.

As the experiment was done offline on a PC, this study considers the running time as a measure of the computational cost of the system. To see the impact of the running time on the energy consumption, this study refers to the total energy E model from Dunkels *et al.* [48]:

$$\frac{E}{V} = I_m t_m + I_l t_l + I_t t_t + I_r t_r + \sum_i I_{c_i} t_{c_i}, \tag{5.4}$$

where V is the supply voltage, and I_x and t_x are the current draw and the running time, respectively, for x, where x can be:

- the microprocessor in normal mode m,
- the microprocessor in low-power mode l,
- the communication device in transmit mode t,
- the communication device in receive mode r, and
- other components c.

The energy consumption of the microprocessor in all modes except the m mode were assumed to be zero ($I_l = I_t = I_r = I_c = 0$), because the simulation is done in a PC using recorded datasets. Assume that there are are two segments S_1 and S_2, where two set of different features $F_1 = \{f_{11},\ f_{12},\ ...,\ f_{1n}\}$ and $F_2 = \{f_{21},\ f_{22},\ ...,\ f_{2n}\}$ are extracted from S_1 and S_2, respectively. These segments refer to periods of time and the associated raw sensor measurements. Then the relative energy between two segments (S_1 and S_2) implemented on the same device is

$$\frac{E_1}{E_2} = \frac{I_{m_1} t_{m_1} V_1}{I_{m_2} t_{m_2} V_2}. \tag{5.5}$$

Because those two segments are assumed to be implemented on the same sensor device, the current draw and the supply voltage are assumed to be the same as well ($I_{m_1} = I_{m_2}$ and $V_1 = V_2$). Then, the energy is proportional to the running time $E \propto t$. This chapter uses a Wilcoxon signed-rank test to evaluate the significance of the improvement in both the detection rate and the computational cost. This method is chosen because the precision, recall, F-score, and computational cost values are not normally distributed. The normality of those data was tested using the Shapiro-Wilk normality test.

5.3 Event-triggered machine-learning approach (EvenT-ML)

The event-triggered machine-learning approach (EvenT-ML) can be described as a finite state machine. This is helpful as it ensures that EvenT-ML can be executed on-line with minimal memory requirements.

5.3.1 Event-triggered machine-learning approach (EvenT-ML) state machine

EvenT-ML consists of four states: **Initial buffer, Peak detection**, **Multi-peak detection**, and **Sample gathering**. EvenT-ML's state machine is shown in Figure 5.3.

Initial buffer: When the system is started, this state is executed once, and collects samples as a buffer. The idea of this buffer is to provide enough samples to be considered as the pre-impact stage, when a sudden fall appears after the system is started. This state uses a timer called the buffer timer (*bft*) with a length of t_{pre}.

Peak detection: This state looks for peaks in the acceleration vector magnitude (a_{vm}). A peak is an a_{vm} that is higher than a threshold τ. If a peak occurs, it is assumed that the subject is active and the state of the system is changed to **Multi-peak detection**.

Multi-peak detection: During a fall, several acceleration peaks can be produced. EvenT-ML assumes that if a fall has occurred then the highest peak corresponds to the moment when the body hits an object (impact stage) [2, 11, 118]. EvenT-ML identifies the alignment of the impact stage by finding the highest peak during a particular length of time t_{mp}. If there is another peak higher than the *recorded peak*, this is taken as the new highest peak and the counter (*mp timer)* of this state is reset. This counter ensures that the length of the impact stage is equal to

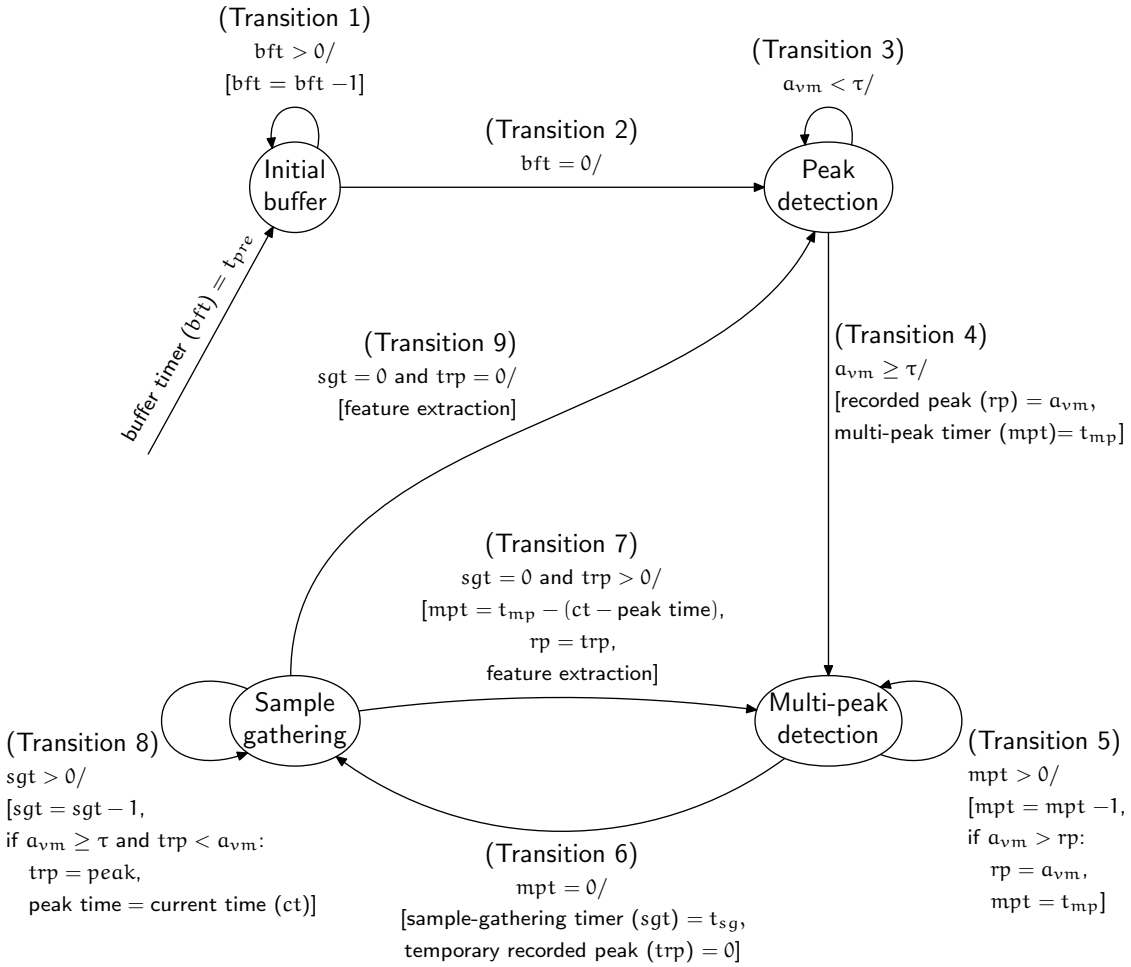


Figure 5.3: EvenT-ML's state machine

t_{mp}. The pre-impact stage is defined as all samples before the *recorded peak,* where the length of this stage is t_p.

Sample gathering: After detecting the highest peak, further samples during a certain amount of time t_{sg} are collected so that a complete fall segment (including pre-impact, impact, and post-impact stages) can be captured. If there is an a_{vm} higher than τ, that a_{vm} value is stored as a *temporary recorded peak* and the *current time* is recorded. Both the *temporary recorded peak* value and its time are updated if another higher peak occurs. The *temporary recorded peak* concept is important in order to avoid missing any peaks, as those peaks could be an indicator of a fall. When the counter for this state has ended, feature extraction is executed. This counter is called the *sg timer*. After performing feature extraction, if the value of the temporary recorded peak is equal to 0, the state is changed to **Peak searching**. Otherwise, the state is changed to **Multi-peak detection** and the *temporary recorded peak* is set as a new *recorded peak*. Then, the *mp timer* is set to $t_{mp} - (\textit{current time} - \textit{peak time})$.

The state machine above produces a segment where:

- all samples during t_{pre} (in seconds) before the highest peak are considered as the pre-impact stage,
- all samples during t_{mp} (in seconds) starting from the highest peak are considered as the impact stage, and
- all samples during t_{sg} (in seconds) after the impact stage are considered as the post-impact stage.

Then the features are extracted from the pre-impact, impact, and post-impact stages (see Subsection 4.4 for more detailed information regarding the features). Because these features are extracted from each stage, the total number of features used is 27. A feature selection is not done in this chapter because one of the purposes of this chapter is to evaluate the effectiveness of using fall stages (EvenT-ML) compared to

without using fall stages (FNSW- and FOSW-based machine-learning approaches) as a basis for feature extraction. The issue regarding the feature reduction is investigated in Chapter 6.

5.3.2 Parameter selection for EvenT-ML

Section 5.3 shows that EvenT-ML has some parameters to define: the pre-impact (t_{pre}), impact (t_{mp}), post-impact (t_{sg}) intervals, and the threshold (τ). This subsection discusses a selection process of those parameters, to maximise the F-score of EvenT-ML. This selection process uses the Cogent dataset because it has more variations, in terms of length, of both falls and ADLs than the SisFall dataset. Using the SisFall dataset for the selection process might not be appropriate as it has a uniform length of fall data (15 seconds). By using a 15-second window, the classifier can predictively achieve the highest F-score (see Chapter 4 for the analysis of the window-size impact on the SisFall dataset). However, the lengths of human activities are unpredictable in real cases, and the result from the SisFall dataset might not be able to represent the performance of the classifier in the real cases. This means that the SisFall dataset might not be appropriate to define the window size. Thus, this section uses the Cogent dataset to select the parameter values of EvenT-ML.

5.3.2.1 Choice of threshold (τ)

The aim of the threshold τ is to ensure that only energetic activities are forwarded for feature extraction, where it is important to prevent the system from extracting features from all possible segments. This prevention mechanism allows the system to reduce the computational cost of the feature-extraction process, as it is clear that extracting complex features can increase the computational cost of the system [92]. Table 5.2 shows thresholds ($1.6g$, $1.8g$, and $1.9g$) that are proposed by some existing studies for detecting an active state.

Table 5.2: Thresholds for detecting active state from existing studies

Reference(s)	Threshold
Abbate *et a*l. [2]	
Ojetola [118]	$1.6g$
Putra *et al.* [127]	
Kau *et al.* [92]	$1.8g$
Chen *et al.* [34]	$1.9g$

Based on the Cogent dataset, the minimum impact-peak value (the highest peak that is produced when the body of the subject hits the ground) of the fall data across the Cogent dataset is $1.8g$. This means that the threshold from Chen *et al.* [34] ($1.9g$) can cause some falls to be not captured for further processing. Thus, this study considers thresholds of $1.6g$ [2, 120, 127] and $1.8g$ [92].

5.3.2.2 Choice of the size of the fall stages

As both the Cogent and SisFall datasets do not provide annotation or video to differentiate between the pre-impact and impact stages, it is difficult to estimate the precise length of those stages. Thus, a 1-second window of t_{pre} was used based on Ojetola's [118] and Aziz [9].

As the pre-impact stage is to be estimated as 1 second and most of the fall data (including both pre-impact and impact stages) from the Cogent dataset span 2 seconds or longer (Figure 4.2), the minimum impact-stage time is 1 second. Ojetola [118] suggests 6 seconds as the impact-stage length to optimise the F-score. Thus, this study considers the use of 1–6 seconds of window for the impact stage.

The post-impact stages from the Cogent dataset are supervised; for example the protocol suggested the subjects to remain lying down for 10 seconds after falling. This causes the length of the post-impact stage to be less natural, as in the real case its size is unpredictable (the victim can lie on the floor for a long period of

time if he/she loses his/her consciousness). Thus, this study evaluates a range of 1–6 seconds for the post-impact stage, where this range is suggested by Banos *et al.* [13].

5.3.2.3 Parameter selection results

This section uses four machine-learning algorithms: CART, k-NN, LR, and SVM tested on the Cogent dataset to select the values of the parameters t_{mp}, t_{sg}, and τ of EvenT-ML in order to maximise its F-score. The following parameters were considered for tuning k-NN, LR, and SVM:

- k-NN with $k = 1, 2$, and 3;
- LR with an inverse regularisation strength (C) of $10^8, 10^9$, and 10^{10};
- SVM with linear, polynomial, and radial-basis-function (RBF) kernels.

To choose the best value of k for k-NN, C for LR (where these parameters are called hyper-parameters), and the best kernel for SVM, the parameter values above were tested, and the ones that give the best F-score were selected. The test was done by evaluating each combination of the parameters values above using LOSOCV, and the F-score of each combination is reported.

Tables 5.3 and 5.4 show the F-scores of EvenT-ML (on average) with $1.6g$ and $1.8g$ thresholds on different impact (t_{mp}) and post-impact (t_{sg}) window sizes using CART, k-NN, LR, and SVM on the Cogent dataset. These results show that increasing either the impact or the post-impact window size does not necessarily increase the F-score. The best window size for both the impact (t_{mp}) and post-impact (t_{sg}) stages is 1 second. In terms of F-score, using $1.8g$ of threshold (when impact and post-impact window sizes are 1 second) gives a significantly better result than using $1.6g$ of threshold (p-value = 0.02). Based on these parameters, Figure 5.4 shows an example of a fall segmentation (with annotation added) produced by EvenT-ML.

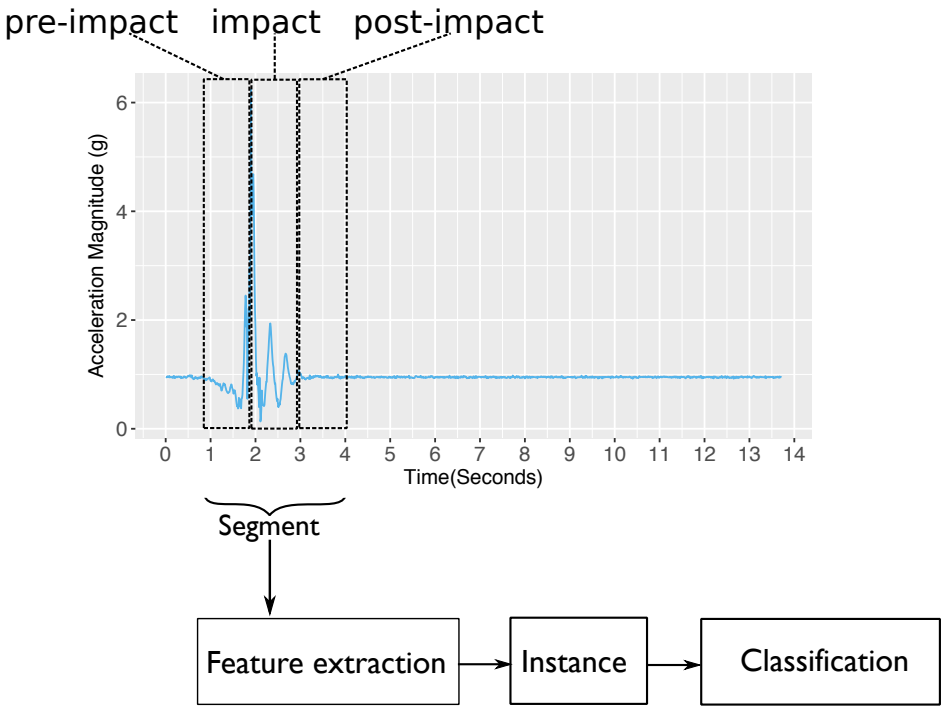


Figure 5.4: A data segment produced by EvenT-ML

With regard to the machine-learning algorithm parameters, Table 5.5 shows F-scores of k-NN-, LR-, and SVM-based classifiers when $t_{mp} = 1$s, $t_{sg} = 1$s and $\tau = 1.8g$. Based on these results, this study chooses the following parameters, as they can give the highest F-score:

- $k = 3$ for k-NN,
- $C = 10^9$ for LR,
- Linear kernel for SVM.

In fact, for the LR-based classifier, there is not a significant difference between using $C = 10^8$, $C = 10^9$, or $C = 10^{10}$, with p-values≥ 0.4. EvenT-ML uses those selected parameters for a comparison analysis with the sliding-window-based (FNSW and FOSW) approaches, CCA [127], and IMPACT+POSTURE [87] in Section 5.4.

Table 5.3: Overall F-score (average and standard deviation) of EvenT-ML with different impact and post-impact window sizes with $\tau = 1.6g$

Post-impact window size (s)	**Impact window size**					
	1 second (%)	**2 seconds (%)**	**3 seconds (%)**	**4 seconds (%)**	**5 seconds (%)**	**6 seconds (%)**
1	94.4±8.7	93.9±8.9	94±9.1	93.7±9.2	93.1±9.7	93.2±10
2	93.8±8.7	94±9.3	93.7±9.4	93.5±9.2	93.4±9.2	93.1±10.2
3	93.8±8.8	93.8±9.4	93.6±9.1	93.7±9	93.4±9.6	93.5±9.9
4	93.4±9.2	93.6±9.5	93.6±9.1	93.4±9.7	93.7±9.2	93.8±9.2
5	93.3±9.2	93.6±9.3	93.6±9.2	93.7±9.4	94±9	94±9.3
6	93.4±9.2	93.7±9.5	93.7±9.2	94.1±8.9	94±9.3	93.8±9.4

Table 5.4: Overall F-score (average and standard deviation) of EvenT-ML with different impact and post-impact window sizes with $\tau = 1.8g$

Post-impact window size (s)	**Impact window size**					
	1 second (%)	**2 seconds (%)**	**3 seconds (%)**	**4 seconds (%)**	**5 seconds (%)**	**6 seconds (%)**
1	94.7±8.4	94±9.1	93.9±9.2	93.7±9.2	92.9±9.9	93.5±9.5
2	94.3±8.7	93.9±9.2	93.7±9.3	93.5±9.1	93.3±9.2	93.1±10
3	94.3±8.5	93.7±9.4	93.6±9.1	93.6±9	93.4±9.6	93.6±9.6
4	93.9±8.9	93.5±9.5	93.7±9	93.6±9.5	93.6±9.3	93.9±9.2
5	93.9±9	93.8±9.1	93.5±9.4	93.8±9.3	93.8±9.1	94±9.2
6	93.8±9	93.8±9.3	93.9±9.3	94.1±9	94±9.3	94.1±9.1

Table 5.5: Average and standard deviation of F-score of k-NN, LR, SVM when $t_{mp} = 1$ s, $t_{sg} = 1$ s and $\tau = 1.8g$ are used on EvenT-ML

Metrics	***k*-NN (%)**			**LR (%)**		**SVM (%)**		
	$k = 1$	$k = 2$	$k = 3$	$C = 10^8$	$C = 10^9$	$C = 10^{10}$	**Linear**	**RBF**	**Polynomial**
F-score	92.7±7.9	91.4±11	94±8.8	97.5±3.3	97.6±3.3	97.5±3.2	95.7±8.2	93.5±10.3	94.6±10.3

5.4 Results and analysis

This section consists of three sub-sections. The first sub-section aims to compare EvenT-ML and both FNSW- and FOSW-based machine learning. The second sub-section analyses the improvement of EvenT-ML compared to CCA and IMPACT+POSTURE. The last sub-section covers the evaluation of EvenT-ML with different sensor placements.

5.4.1 A comparison between EvenT-ML and sliding-window techniques

The aim of this comparison is to investigate the effectiveness of EvenT-ML in increasing the performance (precision, recall, and F-score) of the machine-learning-based approach, while decreasing its computational cost (running time).

5.4.1.1 Cogent dataset

This sub-section conducts a comparison study between EvenT-ML, Fixed-size Non-overlapping Sliding Window (FNSW), and Fixed-size Overlapping Sliding Window (FOSW). For a fair comparison, 3 seconds of FNSW and FOSW (with 25%, 50%, 75%, and 90% of data overlap) were compared to EvenT-ML. The machine-learning algorithms use the parameters that are selected in the previous section ($k = 3$ for k-NN, $C = 10^9$ for LR, and linear kernel for SVM).

Tables 5.6–5.8 provide results of the experiment in terms of precision, recall, and F-score. In terms of precision, EvenT-ML performs better than FNSW and FOSW regardless of the machine-learning algorithm. This implies that EvenT-ML has fewer false alarms than both FNSW and FOSW. EvenT-ML has better recall than FNSW and FOSW when LR and SVM are used to build the classifier, while FNSW achieves a lower detection rate than FOSW regardless of its overlap size and the machine-learning algorithm.

Overall, EvenT-ML is able to significantly outperform both FNSW and FOSW by having the best F-score (harmonic mean between precision and recall) in all cases regardless of the machine-learning algorithm. Compared to both FNSW and FOSW, EvenT-ML can achieve a significantly better F-score (p-values $\leq 2.9 \times 10^{-6}$). These results show that EvenT-ML can maintain the balance between the number of false alarms and undetected falls better than FNSW and FOSW.

Table 5.9 shows the number of segments, the running time for each segment, and the total computational cost for $Stage_1$ of FNSW, FOSW, and EvenT-ML. EvenT-ML has significantly fewer segments over which it runs feature extraction than FNSW and FOSW. Per segment, EvenT-ML has a larger running time than FOSW and FNSW, as it extracts features from three stages (pre-impact, impact, and post-impact). Although having a higher computational cost per segment than FNSW and FOSW due to a higher number of features, EvenT-ML still has a total lower computational cost ($Stage_1$). This is because EvenT-ML produces fewer segments than FNSW- and FOSW-based machine-learning approaches. Compared to FNSW, EvenT-ML can reduce the total computational cost by a factor of 8, and of up to 80 compared to FOSW. Based on the Wilcoxon signed-rank test, EvenT-ML is able to achieve a significantly lower computational cost than both FNSW and FOSW, with p-values $\leq 4.9 \times 10^{-27}$. Note that the running-time values can be different for every iteration. This is because the running-time calculation can be interrupted by other processes in the background. Since the feature extraction was done several times for all subjects, the standard deviation of the running time of the feature extraction is provided in Table 5.9 to show the distribution of the running time. Also, for this comparison, both the running time and the number of segments produced are considered since the number of segments is not affected by the interruption.

Table 5.6: Average and standard deviation of precision (%) of machine-learning approaches on Cogent dataset

Segmentation technique	**CART**	***k*-NN**	**LR**	**SVM**
EvenT-ML	91.4±8.8	95.6±7.2	97.2±4.1	97.2±5.6
FNSW	44.5±9.4	72.6±12.6	91.7±11.3	94.5±8.4
25%-FOSW	41.2±8.2	66.3±13.3	90.1±11.6	93.2±9.5
50%-FOSW	35.3±6.4	59.3±12.7	89.8±11.4	93.4±9.8
75%-FOSW	27.8±4.3	44.7±9.3	88.6±11.4	93.1±10.2
90%-FOSW	21.4±2.4	29.4±5.4	87.5±12.3	92.5±10.8

Table 5.7: Average and standard deviation of recall (%) of machine-learning approaches on Cogent dataset

Segmentation technique	CART	k-NN	LR	SVM
EvenT-ML	92.4±11.2	93.2±12.1	98.1±3.8	94.7±11
FNSW	92.4±10.1	89.9±13.9	87.3±17.9	83.9±20.7
25%-FOSW	94.6±7.7	91.1±12.8	90.4±15.4	85.6±20.2
50%-FOSW	97±4.9	95.2±8.8	92.9±13.1	85.7±20.1
75%-FOSW	98.8±3.8	98.1±5.3	94.4±10.8	86.3±20
90%-FOSW	99.7±1.5	99.7±1.5	95.2±10.7	86.6±19.8

Table 5.8: Average and standard deviation of F-score (%) of machine-learning approaches on Cogent dataset

Segmentation technique	CART	k-NN	LR	SVM
EvenT-ML	91.6±9.3	94±8.8	97.6±3.3	95.7±8.2
FNSW	59.6±9.5	79.7±11.5	88.3±13.3	87.2±15.7
25%-FOSW	56.9±8.2	76.1±11.8	89.2±11.6	87.9±15.5
50%-FOSW	51.5±7.1	72.5±10.7	90.6±10.4	87.9±15.3
75%-FOSW	43.2±5.3	60.9±9.5	90.8±9	88.2±15.5
90%-FOSW	35.2±3.2	45.2±6.4	90.5±9.3	88±15.4

Table 5.9: Average and standard deviation of number of segments, computational cost for each segment, and running time for each segmentation technique on a subject (Cogent dataset)

Segmentation technique	Number of segments	Computational cost for each segment (ms)	$Stage_1$ (ms)
EvenT-ML	38.4±9.9	0.9±0.1	34.3±8.7
FNSW	429.4±84.8	0.6±0.2	269.5±54.6
FOSW 25%	572.3±113.1	0.6±0.6	367.3±79.9
FOSW 50%	858.2±169.6	0.6±0.7	555.2±113.1
FOSW 75%	1715.8±339.2	0.6±0.6	1098.9±219.8
FOSW 90%	4288.8±848.4	0.6±0.4	2528.3±498

5.4.1.2 SisFall dataset

A similar setup of EvenT-ML is used for this dataset (τ =1.8g, 1-second windows for the pre-impact, impact, and post-impact stages). Moreover, the same parameters are applied to the machine-learning parameters. Tables 5.10–5.12 show the precision, recall, and F-score of EvenT-ML and sliding-window-based approaches on different machine-learning algorithms.

EvenT-ML is able to achieve a better performance than both FNSW and FOSW in terms of precision. However, EvenT-ML achieves lower recalls than both the FNSW- and FOSW-based approaches. In general, by looking at the F-score, EvenT-ML is able to significantly outperform both FNSW- and FOSW-based machine-learning approaches regardless of the machine-learning algorithm (p-values $\leq 4.9 \times 10^{-5}$).

Table 5.13 shows the average number of segments produced, the running time for each segment, and the total computational cost ($Stage_1$). EvenT-ML is able to use significantly fewer segments than both FNSW and FOSW (p-values $\leq 4.3 \times 10^{-5}$). Similarly to the results from the Cogent dataset, EvenT-ML has a larger computational cost for each segment than both FNSW and FOSW on the SisFall dataset. This is because EvenT-ML needs to extract features from each fall stage. In terms of the $Stage_1$ cost, EvenT-ML achieves significantly less computational cost than both FNSW (by a factor of 2) and FOSW (by a factor of up to 20) with p-values $\leq 9.5 \times 10^{-13}$.

5.4.2 A comparison between EvenT-ML and the cascade-classifier approach (CCA)

Algorithm 5.1 shows the process of the cascade-classifier approach that was developed by Putra *et al.* [127]. This technique uses a 2-second FNSW to check the state of the body (**Check_act**). If the highest peak occurring during that 2-second

Table 5.10: Average and standard deviation of precision (%) of machine learning approaches on SisFall dataset

Segmentation technique	**CART**	***k*-NN**	**LR**	**SVM**
EvenT-ML	83.5±4.8	87.5±5.1	88.4±5.1	90.3±5
FNSW	52.3±1.3	53.5±2.1	51.9±1.7	50.7±0.6
25%-FOSW	51.1±1.3	52.3±2.1	51.4±1.7	50.3±0.5
50%-FOSW	49.5±0.6	50.2±0.9	50.6±1.4	49.9±0.4
75%-FOSW	48.9±0.3	49.2±0.5	50±0.8	49.7±0.3
90%-FOSW	48.7±0	48.8±0.1	49.6±0.3	49.6±0.2

Table 5.11: Average and standard deviation of recall (%) of machine learning approaches on SisFall dataset

Segmentation technique	**CART**	***k*-NN**	**LR**	**SVM**
EvenT-ML	92.5±7.7	94.5±5.8	94.6±4.8	92.7±8.9
FNSW	99.8±0.9	99.9±0.3	99.9±0.3	100±0
25%-FOSW	99.9±0.3	100±0	100±0	100±0
>25%-FOSW	100±0	100±0	100±0	100±0

Table 5.12: Average and standard deviation of F-score (%) of machine learning approaches on SisFall dataset

Segmentation technique	**CART**	***k*-NN**	**LR**	**SVM**
EvenT-ML	87.7±5.5	90.7±4.2	91.3±3.3	91.1±5.2
FNSW	68.6±1.1	69.7±1.8	68.3±1.4	67.3±0.5
25%-FOSW	67.6±1.1	68.6±1.8	67.9±1.5	66.9±0.5
50%-FOSW	66.2±0.5	66.8±0.8	67.2±1.2	66.5±0.3
75%-FOSW	65.7±0.3	66±0.4	66.7±0.7	66.4±0.3
90%-FOSW	65.5±0	65.6±0.1	66.3±0.3	66.3±0.2

Table 5.13: Average and standard deviation of number of segments, computational cost for each segment, and running time for each segmentation technique on a subject (SisFall dataset)

Segmentation technique	**Number of segments**	**Computational cost for each segment (ms)**	$Stage_1$ **(ms)**
EvenT-ML	323.7±38.7	1.5±0.1	473.5±53.7
FNSW	873±0	1.2±1.8	1000±35
FOSW 25%	1163.9±0.3	1.2±0.7	1300±48
FOSW 50%	1745±0	1.1±1.9	2100±51
FOSW 75%	3489.9±0.3	1.1±1.5	3800±25
FOSW 90%	8723.8±0.7	1.1±1.1	94500±52

Table 5.14: State and transition occurrences (average and standard deviation) on each subject on both datasets

Dataset	**Transition 5's counter**
Cogent	36.5 ±8.3
SisFall	159.5±18.5

window is higher than the threshold of 1.6g, the state of the body is considered as active. Then, when the state of the body is active, the 1 second before the highest peak and the 11 seconds after the peak are captured as a segment for feature extraction. Figure 2.5 shows an illustration of the segments produced by CCA. It is hard to determine the beginning and the end of each fall stage on those two segments because they are not aligned. This section shows the improvement that is caused by solving that multi-peak issue.

To investigate the existence of the multi-peak issue on the Cogent and SisFall datasets, a counter is placed in Transition 5 (see Figure 5.3). Table 5.14 shows the number of multi-peak occurrences when $t_{pre} = 1$s, $t_{mp} = 1$s, $t_{sg} = 1$s and $\tau = 1.8$g for both datasets. Table 5.15 shows that, compared to CCA, EvenT-ML is able to achieve improved precision and recall. In terms of F-score, EvenT-ML can achieve significantly better performance than CCA (p-values $\leq 5.6 \times 10^{-5}$). EvenT-ML is also able to outperform CCA in terms of precision and recall, regardless of the machine-learning approach, on the SisFall dataset (Table 5.15). In general, based on the F-score, EvenT-ML achieves a significantly better performance than CCA (p-value $\leq 2.4 \times 10^{-4}$).

The results above (from both the Cogent and SisFall datasets) show that appropriately handling the multiple acceleration peaks leads to a significant improvement in the classification performance (in terms of precision, recall, and F-score).

Algorithm 5.1 Cascade-classifier approach

```
i ← 1 second                                   ▷ index for the active-state-checker window
vm ← record of acceleration vector magnitude
instances ←[]
while datastream do
    seg ←vm[i : i + 2 seconds]
    if any samples in seg >1.6 then
        peak ← the highest peak in seg
        segfeat ← vm[i − 1 second : i + 12 seconds]▷ window size = 12 seconds
        instance ← Feature_Calculation(segfeat)
        instances ← add(instance)
        i = i + 2 seconds
    end if
end while
```

Table 5.15: Cascade-classifier approach (CCA) on different machine-learning algorithms

Approach	**Precision (%)**		**Recall (%)**		**F-score (%)**	
	Cogent	**SisFall**	**Cogent**	**SisFall**	**Cogent**	**SisFall**
CCA+CART	86.6±12.3	82.9±3.9	84.6±14.9	84.6±11.3	83.9±7.1	83.3±7.5
CCA+k-NN	83.1±11.5	81.1±4.2	88.8±13.5	87.7±9.5	84.5±6.5	84.1±6.4
CCA+LR	89.6±6.9	86.3±5	89.3±15.6	83.8±13.8	84.6±9.8	84.4±10
CCA+SVM	87.2±8.3	83.1±4.9	88.5±15.9	86.1±14.7	84±10.3	83.8±10.2

5.4.3 A comparison between EvenT-ML and IMPACT+POSTURE's performance

Although EvenT-ML applies a threshold, its approach is different from IMPACT+POSTURE (threshold-based approach). The main difference between EvenT-ML and IMPACT+POSTURE is that EvenT-ML utilises a machine-learning algorithm to build the classifier, while IMPACT+POSTURE uses manually defined thresholds to build the classifier.

Compared to IMPACT+POSTURE using the Cogent dataset, EvenT-ML is able to achieve a significantly better precision (p-values $\leq 2.4 \times 10^{-8}$), except when Event-ML uses CART to train the classifier (p-value = 0.8). In terms of recall, EvenT-ML can achieve better performance than IMPACT+POSTURE, regardless of the machine-learning technique. In general, by looking at the F-score, EvenT-ML is able to outperform IMPACT+POSTURE regardless of the machine-learning algorithm, with p-values ≤ 0.03.

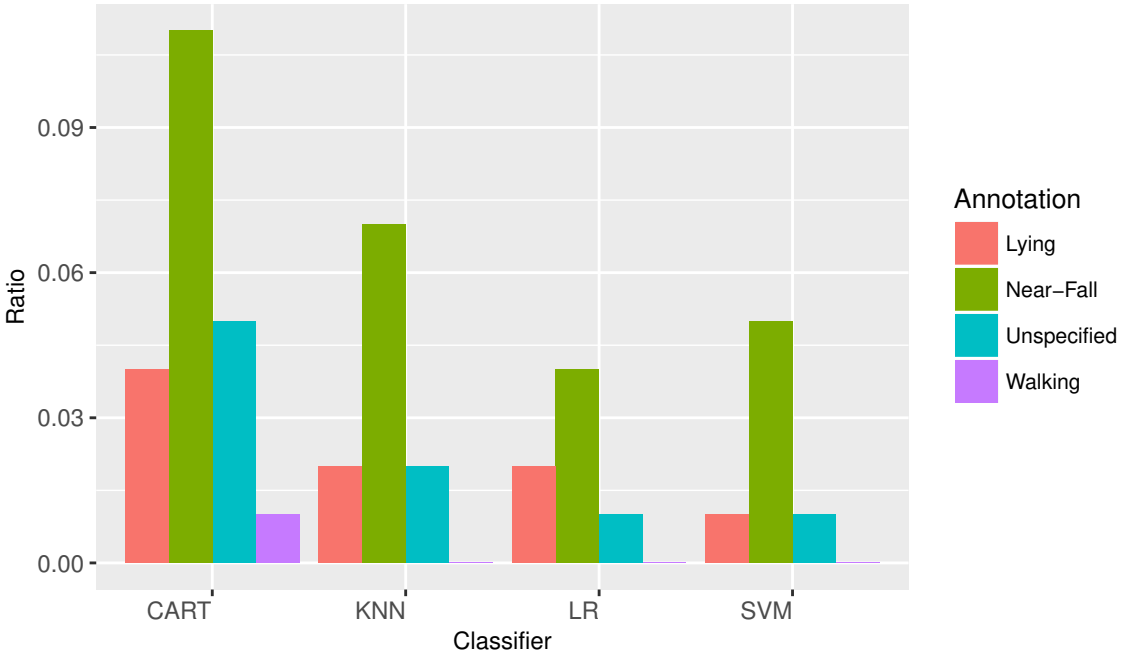
Although EvenT-ML achieves less recall than IMPACT+POSTURE on the SisFall dataset, the EvenT-ML precision is significantly higher than with IMPACT+POSTURE (p-values $\leq 4.3 \times 10^{-5}$). In terms of F-score, EvenT-ML is able to significantly outperform IMPACT+POSTURE (p-values $\leq 4.9 \times 10^{-5}$). These results show that EvenT-ML is able to reduce the number of false alarms, while still maintaining a recall comparable to IMPACT+POSTURE. Also, these results show that the machine-learning-based approach can significantly outperform the threshold-based approach, regardless of the machine-learning algorithm, when the data are correctly segmented based on fall stages (pre-impact, impact, and post-impact). Since Bagala *et al.* [10] show that IMPACT+POSTURE algorithm does not give a better detection rate (in terms of sensitivity and specificity) than algorithms proposed by Bourke *et al.* [25] and Chen *et al.* [34], future work of this thesis is to evaluate algorithms proposed by Bourke *et al.* and Chen *et al.* on

publicly-accessible datasets and compare them with EvenT-ML.

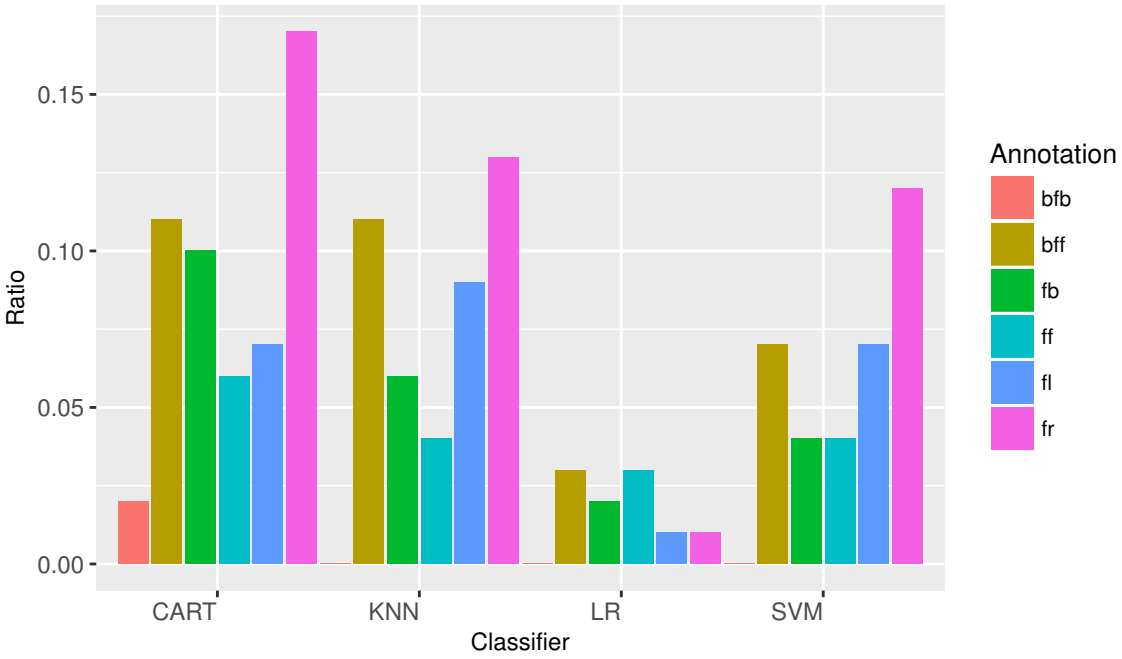
5.4.4 Analysis and discussion of EvenT-ML's performance on the Cogent and SisFall datasets

Figures 5.5a and 5.5b show the false-positive and false-negative ratios for each non-fall activity and fall, respectively (see formulas (5.1) and (5.2) to calculate the false-positive and false-negative ratios, respectively). Figure 5.6b shows the ratio of false negatives for each type of fall of the SisFall dataset. With regard to false alarms, Figure 5.6a shows the false-positive ratio of each non-fall activity from the SisFall dataset. The near-fall is the activity that produces the highest number of false alarms (Figure 5.5a), while fall-to-the-right-side becomes the hardest fall to detect from the Cogent dataset for most of the classifiers (Figure 5.5b).

For the SisFall dataset, the hardest fall to detect is fall-forward while sitting (F13). Figure 5.6b shows the ratio of false negatives for each type of fall of the SisFall dataset. With regard to false alarms, different classifiers find different types of activities hard to detect (Figure 5.6a). These results imply that a near-fall shares similar features with a fall, while fall-to-the-right-side and fall-forward-while-sitting (F13) share similar features with non-fall activities. Lee *et al.*'s study [100] shows that the classifier often misclassifies near-fall as a fall mainly because it accompanies an abrupt movement, where this abrupt movement is similar to a fall. Due to the limited information (for example: videos of all of the subjects performing falls, a subject's preferred hand, or any preventing actions when the subject senses they are about to fall) and a complex interaction between fall dynamics and the machine-learning algorithm used, a further investigation regarding the fall or activity that produces the highest false negative or false positive cannot be thoroughly made. This is a limitation of this study.

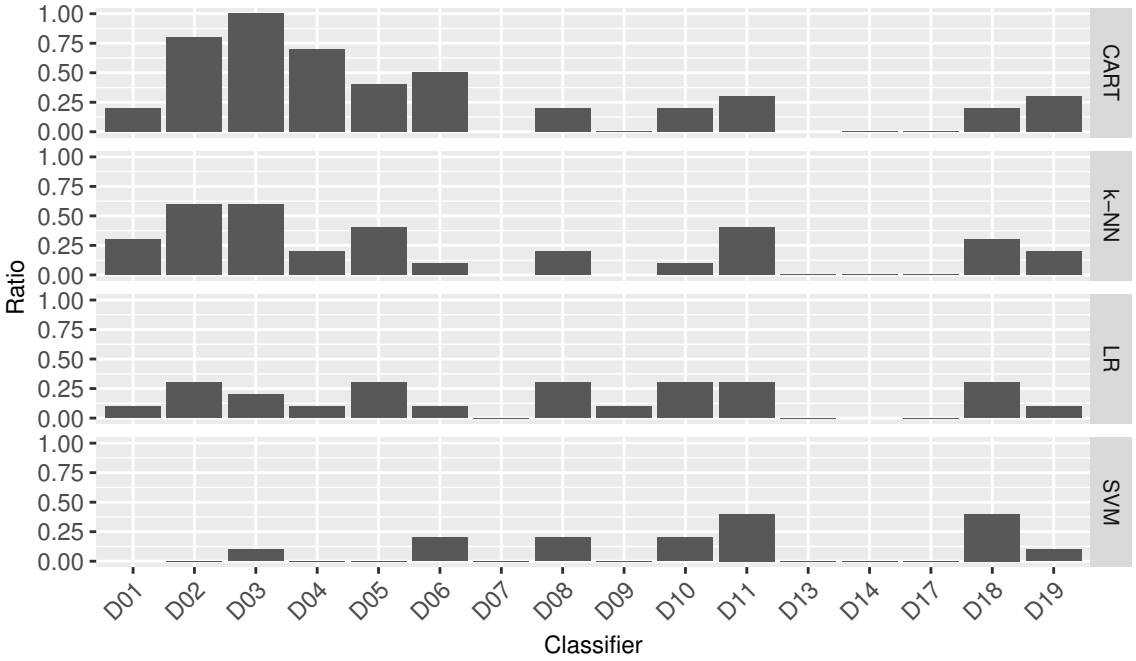


(a) False-positive ratio

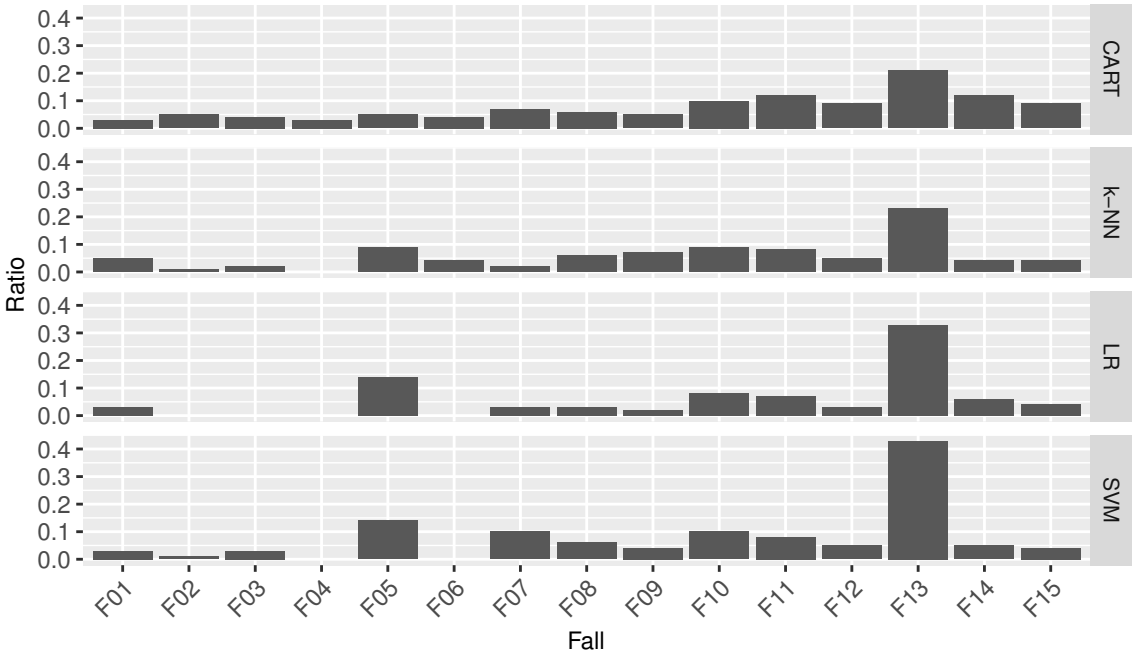


(b) False-negative ratio. (1) bfb = blindfolded fall backward; (2) bff = blindfolded fall forward; (3) fb = fall backward; (4) ff = fall forward; (5) fl = fall to the left side; and (6) fr = fall to the right side.

Figure 5.5: The ratio of false alarms/misclassifications from the Cogent dataset



(a) False-positive ratio



(b) False-negative ratio

Figure 5.6: The ratio of false alarms/misclassifications from the SisFall dataset (see Table 3.3 for the activity index)

5.4.5 A comparison between different placements of the sensor

This subsection uses only the Cogent dataset, because the SisFall dataset only has sensors placed on the subjects' waists. Table 5.16 shows the precision, recall, and F-score of EvenT-ML on each placement. These results show that the chest is the best place to put a sensor to get the optimum detection rate. It is followed by the waist and thigh, respectively. This result is supported by a study from Gjoreski *et al.* [64]. Their study shows that the chest and waist are better places to put the sensor, where the chest placement has a small advantage over the waist placement, though, the waist can give more comfort to the user [163].

5.4.6 Performance analysis using the hold-out technique

Since the performance of EvenT-ML in Tables 5.6–5.8 are results of the parameter tuning of EvenT-ML on the Cogent dataset, this can make the results are biased because some information may "leak" during the process [116]. Thus, this section provides a comparative analysis of EvenT-ML, CCA, IMPACT+POSTURE, FNSW-, and FOSW-based approaches using the hold-out technique. In this case, the Cogent dataset is used for the training set and the SisFall dataset is used for the testing set. For this analysis, only subjects that have data from the waist sensor from the Cogent dataset (see Table 3.1 for the Cogent dataset's subjects body profile) is used because the SisFall dataset only has waist-sensor placement. The results of this experiment are less biased since the testing data have never been used to tune the parameters. Also, this hold-out technique is recommended by Igual *et al.* [80] and Aziz *et al.* [8] to avoid biased results. Table 5.17 shows the hyper parameters used for this sub-subsection. The hyper parameters for FNSW- and FOSW-based approaches are defined in the previous chapter (see Subsection 4.5.1).

Table 5.18 shows the precision, recall, and F-score of the evaluation using the

Table 5.16: Average and standard deviation of precision, recall, and F-score (%) for each placement using machine-learning algorithms

(a) Precision

Placement	**CART**	***k*-NN**	**LR**	**SVM**
Chest	95.3±7.8	96.2±6.3	97.1±5	98.2±3.5
Waist	90.6±9.6	90.8±8.7	96.4±9	94.3±9.9
Thigh	74.6±11.2	77.6±13.6	86.1±9.2	86.8±8.6

(b) Recall

Placement	**CART**	***k*-NN**	**LR**	**SVM**
Chest	96±8.9	94.4±10.6	97.2±6.1	96±10.5
Waist	91.7±10.5	90.5±14.3	94.8±7.7	92.9±12
Thigh	79.8±8.9	72.6±12.6	85.7±14.9	78.6±18.3

(c) F-score

Placement	**CART**	***k*-NN**	**LR**	**SVM**
Chest	95.2±6.4	94.8±6.7	97±3.8	96.9±7.3
Waist	90.4±6.7	89.8±9.3	95.2±6.4	92.8±8.6
Thigh	76.6±8.3	74.2±10.5	85.3±10.6	81.6±13.2

Table 5.17: Hyper parameters used for performance evaluation using the hold-out technique.

Fall-detection approaches	Window size	Machine-learning algorithm	Machine-learning hyper parameter
EvenT-ML	3 seconds	LR	$C = 10^9$
FNSW	2 seconds	LR	$C = 10^9$
FOSW	2 seconds	SVM	kernel = rbf
CCA	12 seconds	LR	$C = 10^9$

Table 5.18: Precision, Recall, and F-score (%) of fall-detection approaches using the hold-out technique

Fall-detection approaches	Precision	Recall	F-score
EvenT-ML	82.5±4.8	79.7±14.8	80.6±10.5
FNSW	74.5±7.3	65.2±18.8	68.8±14.1
FOSW	75±5.2	75.4±16.4	74.7±11
CCA	0.3±0.8	0.2±0.7	0.2±0.7
IMPACT+POSTURE	77±4.6	67.4±10.3	71.5±6.9

hold-out technique. These results show that EvenT-ML can give a better precision, recall, and F-score on average than FNSW-, FOSW-based, CCA, and IMPACT+POSTURE approaches. Based on Wilcoxon signed-rank test, EvenT-ML can achieve a significantly better F-score than FNSW- (p-value= 4.3×10^{-5}), FOSW-based (p-value= 1.1×10^{-4}), CCA (p-value= 4.3×10^{-5}), and IMPACT+POSTURE (p-value= 1.1×10^{-3}). Furthermore, these results also show that using IMPACT+POSTURE algorithm can achieve similar F-score to FNSW- (p-value= 0.5) and FOSW-based approaches (p-value= 0.2). These results show that when different datasets are used for training and testing, the precision, recall, and F-score decrease dramatically. This trend is also found in Igual et al.'s study [80].

5.4.7 Performance analysis on the FARSEEING dataset

This subsection examines the performance of EvenT-ML on the FARSEEING dataset using only data from the waist, as chest placement is not available from the FARSEEING dataset and the waist placement is the second-best placement (see Table 5.16).

The main aim of these evaluations is to evaluate whether data from young and healthy subjects can be used to train the classifier, to detect falls in older people. Tables 5.19– 5.21 show precisions, recalls, and F-scores of EvenT-ML tested on the FARSEEING dataset. These results show that using the Cogent dataset to train the classifier can give better F-scores than using the SisFall dataset.

Table 5.19: Average and standard deviation of precision of EvenT-ML tested on the FARSEEING dataset

Dataset used for training	**CART (%)**	***k*-NN (%)**	**LR (%)**	**SVM (%)**
Cogent (hold-out)	67.2±39.1	56.7±49.5	71.7±43.2	66.7±48.8
SisFall (hold-out)	31.4±30.5	40.4±39.1	30.4±31.4	45.9±38.1
FARSEEING (LOSOCV)	83.3±36.2	63.3±44.2	62.2±40.6	66.7±48.8

Overall (in terms of F-score), there is no significant difference between using the Cogent dataset (evaluated using the hold-out technique) to train the classifier and using the FARSEEING dataset (evaluated using LOSOCV technique), with p-values ≥ 0.5 when EvenT-ML uses CART, k-NN, or LR to build the classifier. In fact, the results are similar when SVM is used to build the classifier. On the other hand, using the SisFall dataset in the training process gives a lower F-score (evaluated using the hold-out technique) than using the FARSEEING dataset (evaluated using LOSOCV technique). The differences are significant when CART or LR is used to build the classifier, with p-value $= 3.2 \times 10^{-3}$ and p-value $= 0.02$, respectively.

The results provided in this section indicate that using the Cogent dataset to train the classifier is as effective as using the real-fall data from the FARSEEING dataset. These results also give an indication that using data from young and healthy subjects in a fall-detection study can represent the performance of the classifier in detecting falls on older people.

Table 5.20: Average and standard deviation of recalls of EvenT-ML tested on the FARSEEING dataset

Dataset used for training	**CART (%)**	***k*-NN (%)**	**LR (%)**	**SVM (%)**
Cogent (hold-out)	86.7±35.2	60 ±50.7	80±41.4	66.7±48.8
SisFall (hold-out)	86.7±35.2	90±28.0	90.0±28	83.3±36.2
FARSEEING (LOSOCV)	83.3±36.2	73.3±45.8	80±41.4	66.7±48.8

Table 5.21: Average and standard deviation of F-scores of EvenT-ML tested on the FARSEEING dataset

Dataset used for training	**CART (%)**	***k*-NN (%)**	**LR (%)**	**SVM (%)**
Cogent (hold-out)	72.7±36.1	57.8±49.5	73.8±41.7	66.7±48.8
SisFall (hold-out)	41.6±29.4	48.2±36.3	39.9±30.7	54.1±36.7
FARSEEING (LOSOCV)	82.2±35.3	66.7±43.6	67.8±39.1	66.7±48.8

5.5 Discussion and limitations

Although this chapter shows that using the Cogent dataset can give comparable results with using the FARSEEING dataset, these results cannot be applied in general since the number of FARSEEING subjects is relatively small (15 subjects). In fact, Table 5.21 shows that the standard deviation of the classifier in all cases are high. This means that the classifier is unstable due to a small number of samples (only 15 falls from 15 subjects). Thus, a larger real-fall publicly-accessible dataset is still important for evaluating the fall-detection approach.

The next limitation of this study is the use of running time to measure the computational cost. The running time of the feature extraction running on the PC can be affected by other processes in the background. As an alternative, this chapter provides the number of segments produced by each segmentation technique to be compared. A bench-top test is intended to be used to measure the real energy consumption of the feature extraction on a real wearable device in the future [28, 63, 154].

Since the Cogent and SisFall datasets are publicly accessible, recent studies from Khan and Taati [95] evaluate their proposed approach using the Cogent dataset, while Sucerquia *et al.* [141] use the SisFall dataset. Khan and Taati's study shows that a one-class-classification-based approach is able to achieve 100% of sensitivity with 0 false alarms on the Cogent dataset, though this study does not report the computational cost of their approach. On the other hand, Sucerquia *et al.*'s study shows that a simple threshold can give a 99% sensitivity and a 99.51% specificity. The results of Khan and Tati and Sucerquiea *et al.*'s studies are obviously better than the results provided in this study. Thus, it indicates that different types of features can increase EvenT-ML's detection rate.

The number of features remains high because features are extracted from each stage. Thus, Chapter 6 provides a genetic-algorithm-based feature-selection tech-

nique aiming at reducing the number of features. With regards to the machine-learning algorithm, different algorithms or parameters can affect the classifier performance. However, the main purpose of this chapter is to evaluate the improvement of the classifier when EvenT-ML is implemented. Thus, a comprehensive study to compare and tune the parameters of the machine-learning algorithms is left for future work.

5.6 Chapter summary

This chapter describes a novel fall-detection approach called event-triggered machine-learning (EvenT-ML). EvenT-ML is described as a state machine to align fall stages (pre-impact, impact, and post-impact) to the acceleration signal, where the feature-extraction process uses these stages as a basis. Two publicly accessible datasets, Cogent and SisFall, were used to evaluate the fall-detection approaches.

Compared to FNSW and FOSW, EvenT-ML achieves significantly better precision and F-score and still can maintain a relatively good recall on both datasets. The best machine-learning technique for EvenT-ML in this study is LR with $C = 10^9$. By using LR, a 97.2% precision, a 98.1% recall, and a 97.6% F -score can be achieved for the Cogent dataset, while for the SisFall dataset an 88.4% precision, a 91.3% recall, and a 91.3% F-score can be achieved. As an additional advantage, EvenT-ML is able to significantly reduce the computational cost of $Stage_1$ for both datasets. EvenT-ML is able to achieve a reduction by a factor of 8 (on average) compared to FNSW and of up to 80 compared to FOSW on the Cogent dataset, while it achieves a reduction by a factor of 2 (on average) compared to FNSW and of up to 20 compared to FOSW on the SisFall dataset. The degree of energy reduction in a real wearable device might be different from the results presented in this study. This is because there are other processes that might increase the energy consumption of the real wearable device. An implementation of EvenT-ML on a real wearable device is

considered as future work.

From these results, this chapter concludes that EvenT-ML is able to strike a balance between precision and recall, which means that EvenT-ML can reduce both false alarms and undetected falls. Compared to CCA and IMPACT+POSTURE, EvenT-ML can achieve a significantly better F-score. In response to **RQ3**- yes, EvenT-ML can significantly improve the detection rate of the classifier when a segment is correctly aligned with the fall stages, where, as an additional advantage, this approach is able to significantly reduce the computational cost of the system. Regarding the sensor placement, the chest gives the highest precision, recall, and F-score compared to waist and thigh. The waist is the second-best placement followed by the thigh.

Another important finding from this chapter is that using the Cogent dataset and EvenT-ML to train the classifier is as effective (not significantly different) as using the real-fall data from the FARSEEING dataset. This finding shows that using young and healthy subjects to train the classifier to detect falls in older people is appropriate, and the results presented in this study can be used as an indication of the classifier's performance in real-case scenarios.

As the number of features remains high, feature selection is needed to reduce the computational cost by reducing the number of features, and to optimise the accuracy by filtering out irrelevant features. The next chapter proposes a genetic-algorithm-based feature selection that can find a subset of features from different sensor placements that can give an optimum detection rate while reducing the computational cost of the system.

Chapter 6

Genetic-algorithm-based feature-selection technique for fall detection (GA-Fade)

6.1 Introduction

The previous chapter provided a novel machine-learning-based approach for fall detection called EvenT-ML. Although EvenT-ML is able to significantly increase the accuracy of the classifier and reduce the computational cost, the number of features being used by this technique remains high. This chapter evaluates a novel genetic-algorithm-based feature-selection technique for fall detection using multiple wearable sensors, where this technique considers F-score, computational cost, and number of sensors used as the selection criteria. Also, this chapter provides a comparative study between wrapper-, filter-based, and embedded techniques to select features for fall-detection application using wearable sensors.

Since filter-based feature-selection techniques ignore features that can give more information when they are used together [158] and use a single criterion to select

features, this chapter considers a wrapper-based feature-selection technique. To select the best subset of features using a wrapper-based feature selection technique from one sensor in this study, an investigation of 2^{27} (134,217,728) possible combinations are needed, because every stage of a fall produces 9 different features and there are 3 fall stages used in this study (pre-impact, impact, and post-impact). Based on Gjoreski *et al.* [64], using more than one sensor placement can give a better accuracy in detecting falls. Thus, if wrapper-based feature selection is used to find a subset of features from 3 sensor placements (chest, waist, and thigh), $\sim 2.4 \times 10^{24}$ possible combinations need to be investigated. Exhaustively evaluating these combinations requires a huge amount of time, which is a disadvantage of the wrapper-based feature-selection technique. Therefore, a heuristic search is often used to reduce the complexity of an exhaustive search in wrapper methods. Genetic algorithms have been shown to be effective in multi-criteria-based feature selection in some studies such as heart disease, cancer, and handwriting recognition [117, 165], as most other feature-selection techniques are not designed to handle multiple selection criteria [165].

The problem of fall-detection approach is not only to improve the detection rate, but also to reduce the computational cost [154]. Wang *et al.* [154] have proposed a multi-criteria-based feature-selection technique, where the detection rate and energy consumption become the selection criteria. The first issue of this technique is that the way a candidate removed feature being selected for each iteration is not clear. Another issue with this technique is that it does not compare the result of the final output with all results of all evaluated combinations. For example, from Table VI of their paper, it shows that the best result is given when f_1, f_3, f_6, and f_8 are used. In fact, a better result is given when f_1, f_2, f_3, f_5, f_6, and f_8 are used.

The main aim of this chapter is to select features that can give a higher detection rate (F-score) and a lower computational cost (optimisation problem). Since the feature dimension of this study is large, the genetic algorithm is chosen to re-

duce the complexity of the brute-force wrapper-based feature-selection technique since the genetic algorithm has been shown to be effective in multi-criteria-based feature selection in some studies such as heart diseases, cancer, and handwriting recognition [117, 165]. Using the genetic algorithm to improve the efficiency of wrapper-based feature selection in searching the optimum subset of features based on multiple criteria for fall detection using wearable sensors has not been investigated. Thus, the contribution of this chapter is a genetic-algorithm-based feature-selection technique for fall detection (GA-Fade) to select sub-features from a large space based on three criteria: detection rate (in terms of F-score), computational cost, and number of sensors used. These criteria are chosen to select features that have lower computational cost, high detection rate, and use as few sensors as possible. Using fewer sensors can increase the user's comfort and reduce the energy consumption [98]. SelectKBest (filter-based feature-selection technique) and Recursive Feature Elimination (embedded feature-selection technique) from the Scikit-learn library are chosen as a comparison.

The structure of this chapter is as follows: Section 6.2 provides an overview of GA-Fade. Section 6.3 explains methods used in this study, while Section 6.4 provides the results and an analysis of the experiment. Section 6.5 provides a discussion and limitations. Section 6.6 concludes this chapter.

6.2 Genetic-algorithm-based feature-selection technique for fall detection (GA-Fade)

In this study, a steady-state genetic algorithm is proposed (Algorithm 6.1). The genetic-algorithm-based feature-selection technique for fall detection (GA-Fade) consists of five main functions: Initial population generation, fitness function, selection function, crossover function, and mutation function.

Algorithm 6.1 A steady-state genetic algorithm

1: Initialise population P
2: Non-increasingly sort individuals in P based on fitness value
3: **while** number of generation $<$ desired generation **do**
4: Select two parents p_1 and p_2 from P
5: $child_1$, $child_2$ = crossover(p_1, p_2)
6: mutation($child_1$, $child_2$)
7: Insert $child_1$ and $child_2$ into P
8: Sort P
9: Eliminate individual with the lowest fitness from P
10: **end while**

6.2.1 Initial population generation

A genetic algorithm is a searching technique to solve optimisation and searching problems, by using genetics as its model [135]. In its process, the genetic algorithm handles a pool of solutions, where each solution is represented as a chromosome. Each chromosome consists of genes, where a gene is a representation of a value that corresponds to the fitness of the solution it represents. The fitness itself shows how good the solution is.

For feature selection using a genetic algorithm, an individual consists of a sequence of features where each feature is represented by a binary number (either 1 or 0) [117]. Values of 1 indicate selected features while values of 0 show removed features. Figure 6.1 and Algorithm 6.2 show an illustration of the encoding of a chromosome and the algorithm of chromosome encoding, respectively.

Before searching for the optimal sequence of features using the genetic algorithm, several initial chromosomes are needed. Algorithm 6.2 shows the process of chromosome generation of the initial population P. This algorithm makes unlikely that there are not two (or more) similar chromosomes in P by comparing the newly gen-

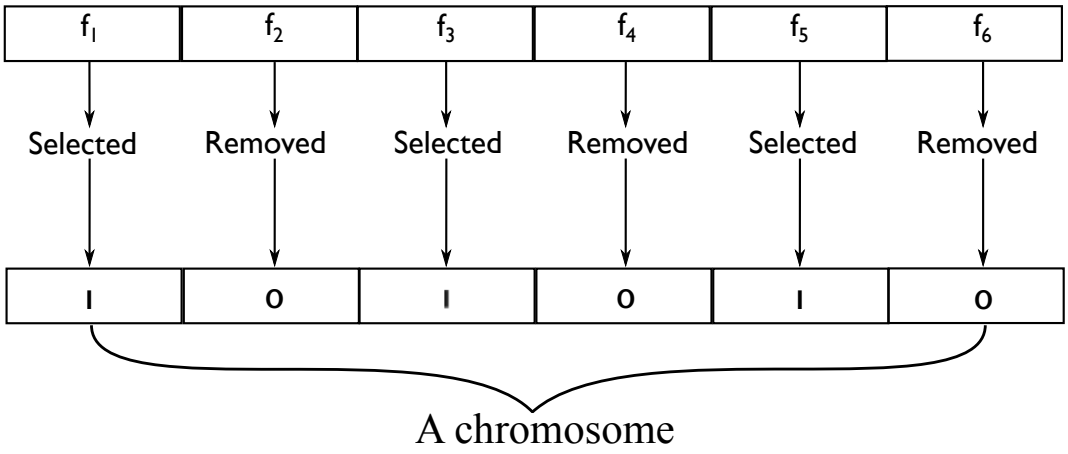


Figure 6.1: Encoding process for GA-based feature selection

erated chromosome with all existing chromosomes in P, to increase the diversity of individuals. This thesis uses 50, 100, and 200 as the number of generations.

Algorithm 6.2 Chromosome encoding

```
for i = 0 to length of a chromosome do
    j = random.uniform()
    if j < threshold then
        digit = 1
    else
        digit = 0
    end if
end for
```

6.2.2 Fitness function

The fitness function has the most important role in the genetic algorithm. This function decides which individual is kept in the population or removed from the population. Based on Venkatraman *et al.* [150], the fitness function of the genetic algorithm can be categorised into three techniques:

1. methods based on penalty functions;

Algorithm 6.3 Controlled initial population generation

```
1: for i = 1 to I do                          ▷ I : Initial size of population P
2:     Generate a chromosome C
3:     while C is exist in P do
4:         Re-generate C
5:     end while
6:     Put C into P
7: end for
```

2. methods based on preference of feasible solutions over infeasible solutions;

3. methods based on multi-objective optimisation.

In this study, a penalty-based function is adopted. To construct the fitness function of our technique, three aspects from Lara and Labrador [98] are considered: level of energy consumption (E), the overall detection rate for fall activities (A), and the number of sensors being used (N). Since all experiments for this study were done offline on a PC and energy is proportional to running time (see the explanation in Section 5.2), the running time is considered to represent E. Because the fall-detection system needs to have low false positives and false negatives [115], the F-score is considered to represent A. The F-score is calculated using formula (2.3). Since the units of variables A, E, and N are different (multi-objective optimisation problem), some weights are implemented [41, 58].

Thus, the fitness function for this study is

$$fitness(x) = w_1.A - w_2.E - w_3.N, \tag{6.1}$$

where:

- x = a chromosome from the population
- A = accuracy in terms of F-score

- w_1 = weight for accuracy
- E = total computational cost of the system for a chromosome
- w_2 = weight for total computational cost
- N = number of sensors chosen
- w_3 = weight for number of sensors used

To compute the total computational cost (E), the following formula is used:

$$E = w_c.t_c + w_w.t_w + w_t t_t, \tag{6.2}$$

where both w_x and t_x are the weight and running time of x, where x can be a chest placement (c), a waist placement (w), or a thigh placement (t). Since the running time might be interrupted by other background processes, the minimum running time is used to get the lowest bound of how fast the machine can run given a code snippet[1].

Reducing the number of sensors (N) is critical for the user's comfort [98]. Also, having fewer sensors (e.g. switching them off when they are not needed) may reduce the energy consumption of the device [28]. Since the E and N values are not bounded, this causes the state space to be infinite. Using an exhaustive search is impossible in this case. Thus, using a genetic algorithm to explore a very large state space can be a better option [65].

The units of A is a percentage (%) and of E is milliseconds (ms). As N is the number of sensors being used, it does not have a unit. Thus, for the implementation, the fitness function is changed into:

$$fitness(x) = (w_1 \times A/100) - (w_2 \times E/2.4) - (w_3 \times N/3),$$

[1]see https://docs.python.org/2/library/timeit.html to see how the Python programming language handles this issue.

where 2.4 and 3 is the least time (in millisecond) of the machine (in this case a PC) to extract all features from all sensors and the total number of sensors used in this study, respectively. w_1 is set to be 1 while w_2 and w_3 is set to be 0.5, where these numbers are empirically chosen. The value of w_1 is set to be higher than w_2 and w_3 because accuracy (A) is more important than energy (E) and number of sensors (N). An equal value is given to w_c, w_w, and w_t, thus $w_c = w_w = w_t = 1$, as the importance of all sensor placements is assumed to be the same. Therefore, the optimum sequence is a sequence that has the highest detection rate with the lowest computational cost and the lowest number of used sensors. Those weights above can be adjusted to fit the needs of the application.

6.2.3 Selection Function

To select parents from the population, a roulette-wheel technique is implemented, adopted from Oh and Lee [117]. This technique is chosen to ensure that fitter chromosomes/individuals have a higher probability of being chosen. The probability $P(i)$ of selecting the ith item from a pool of n items is weighted more highly for lower numbered items according to:

$$P\left(i\right) = q\left(1-q\right)^{i-1} / \left(1-\left(1-q\right)^{n}\right) \tag{6.3}$$

where $q = 1/4$ is the probability of selecting the first item for infinite n [117]. The form of this equation is set to ensure that it sums to unity $\sum_{1 \leq i \leq n} P(i) = 1$ while decreasing the probability geometrically for subsequent items $P(i+1) = (1-q)P(i)$.

It is possible to find an inverse of the cumulative probability distribution to help generate random numbers sampled from this distribution. Specifically, setting $A = \left(1-\left(1-q\right)^{n}\right)$

$$Q(k) = \sum_{0 \leq i \leq k} q\left(1-q\right)^{i-1} / A = 1/A\left(1-\left(1-q\right)^{k}\right) \tag{6.4}$$

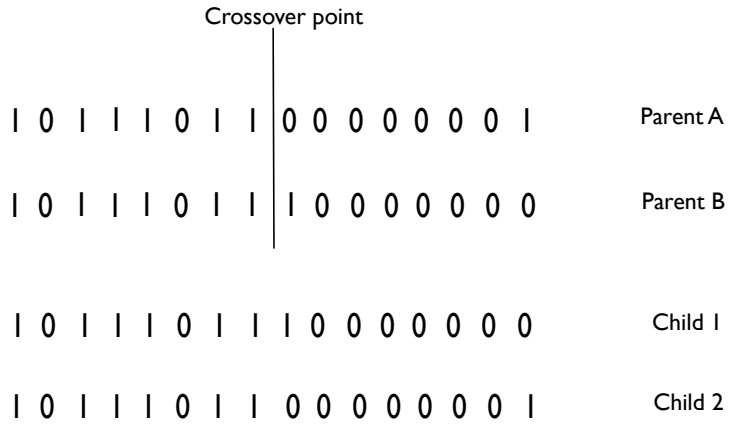


Figure 6.2: An illustration of a one-point crossover operator between parents for one sensor

which has a simple inverse of,

$$k(Q) = \lceil \log(1 - AQ) / \log(1 - q) \rceil \tag{6.5}$$

6.2.4 Crossover function and mutation function

For the crossover function, this study uses a single-point crossover function. Figure 6.2 shows an illustration of the one-point crossover function. For the mutation function, this study adopts a mutation function from [117]. An algorithm for mutating the new individual is shown in Algorithm 6.4. The mutation rate that is used in this study is 0.1, which is adopted from Oh *et al.* [117]. The next sub-section provides the experimental method of this study.

6.3 Method

6.3.1 Experimental Method

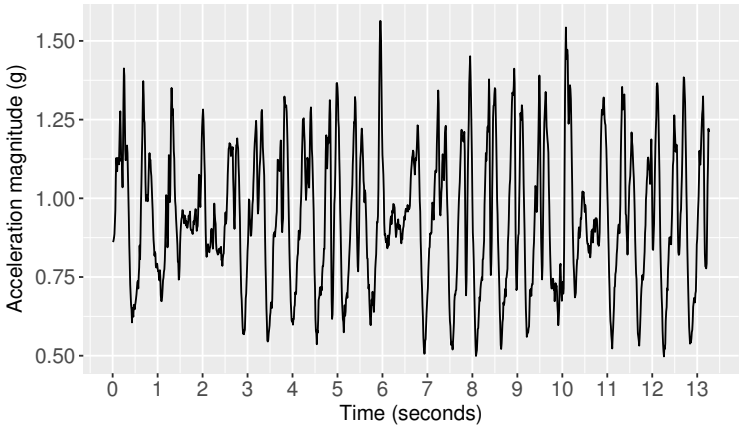
The Cogent dataset is used for this chapter, as it has 3 different sensor placements, and EvenT-ML is used to build the classifier. Only 18 subjects are used in this chapter since they have sensors strapped on their chest, waist, and thigh. Information about the body profile of the subjects is provided in Table 3.1. Because

Algorithm 6.4 Mutation process

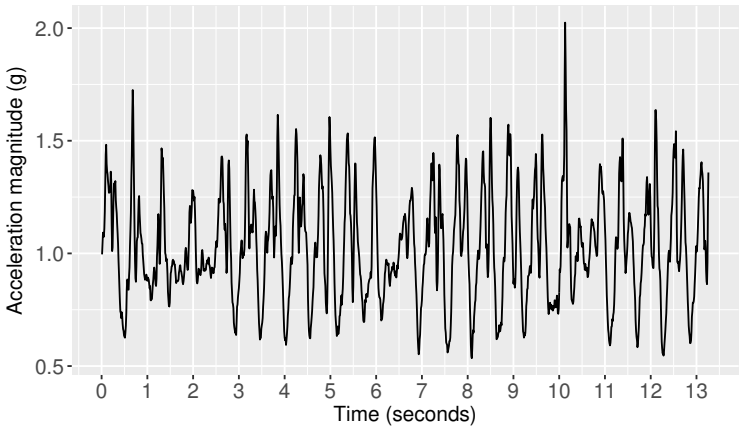
```
for each gene g in the chromosome do
    Generate a random number r within [0, 1]
    if g = 0 and r < p then
        g = 1
    else if g = 1 and r < p then
        g = 0
    end if
end for
```

EvenT-ML extracts the features (see subsection 4.4 for more detailed information regarding the features) based on signal peaks, the number of instances produced from each placement is different for some activities. Figure 6.3 shows examples of acceleration-vector-magnitude signals of a walking activity from chest, waist, and thigh sensors. This figure shows that the thigh sensor produces more peaks, where those peaks are higher than the threshold ($1.8g$), compared to the chest and waist sensors. The chest sensor produces no peak higher than $1.8g$. This makes the thigh sensor have more instances than chest and waist, so that fusion of the feature values between the three sensor placements for feature selection cannot be done directly. Therefore, a data imputation is done, so that all sensors have an equal number of instances. Since an activity is iterated more than one time for each subject, the imputation is done by using the median value of the samples from the same activity. The median value is used for the imputation process because the feature values are not normally distributed. An illustration of this process is shown in Figure 6.4. Since each subject produces a different number of peaks, the number of instances generated for each subject varies.

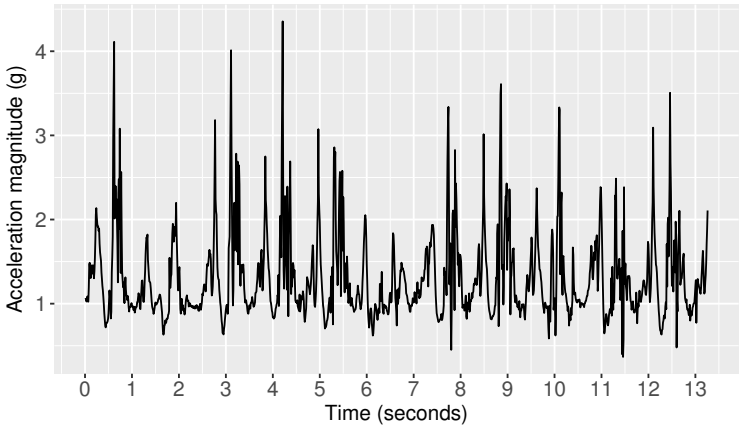
The data were processed offline using a PC. To train the classifier, LR with the inverse regularisation strength (C) of 10^9 from the Scikit-learn library [125] was used.



(a) Chest



(b) Waist



(c) Thigh

Figure 6.3: Acceleration vector magnitude signals for a walking activity from chest, waist, and thigh sensors taken from a subject.

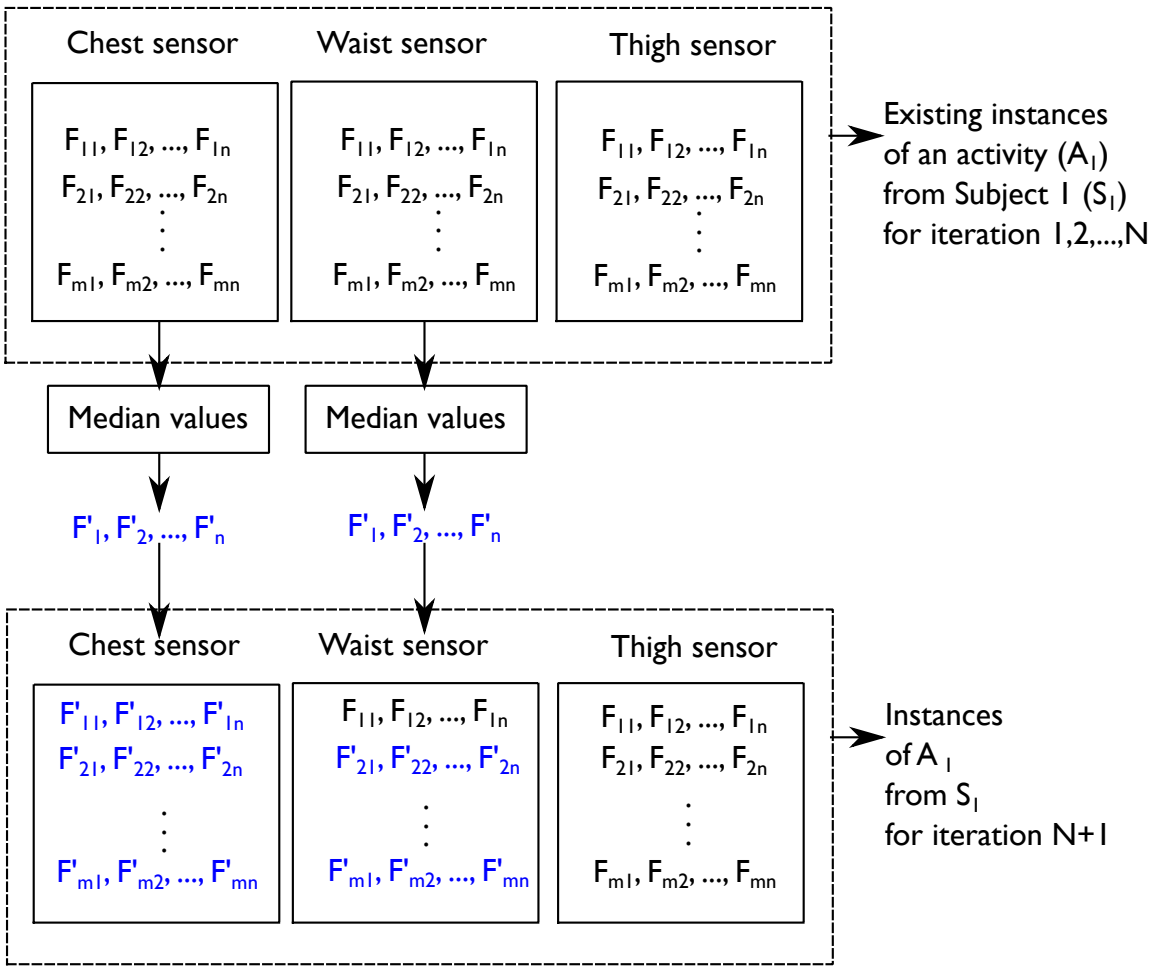


Figure 6.4: Data imputation process

Table 6.1: Average and standard deviation of number of features selected by GA-Fade with different initial population sizes and different numbers of generations

Number of generations	**Size of the initial population (P)**		
	40	**60**	**80**
50	33.1±4.0	33.6±4.5	33.4±4.5
100	28.7±4.3	31.5±3.3	32.3±4.5
200	25.6±3.5	26.4±5.1	25.6±5.5

This algorithm is shown in the previous chapter to be more effective in training a classifier for detecting falls than the other machine-learning algorithms. LOSOCV was used in this experiment. This chapter implements SelectKBest (filter-based feature selection) and Recursive Feature Elimination (embedded feature selection) techniques as a comparison. The significance of the improvement in the detection rate was evaluated using a Wilcoxon signed-rank test with a significance level of 0.05 ($\alpha = 0.05$).

To avoid biased results, a validation step from Nowotny [116] is adopted. Figures 6.5 and 6.6 show the validation steps for the GA-Fade and filter-based methods (SelectKBest and RFE).

6.3.2 SelectKBest feature-selection technique

SelectKBest is a library for feature selection from Scikit-learn using the Python programming language [125]. This library represents one of the filter-based methods, where this technique gives a score to each feature, then non-increasingly ranks them. The number of selected features (K) is defined by the user. A score function is used to measure each feature. This study investigates several score functions: chi-square test (*chi2*) and ANOVA F-value (*f_classif*). In this study, K is equal to 30, as the number of selected features by GA-Fade is about 30 (Table 6.1).

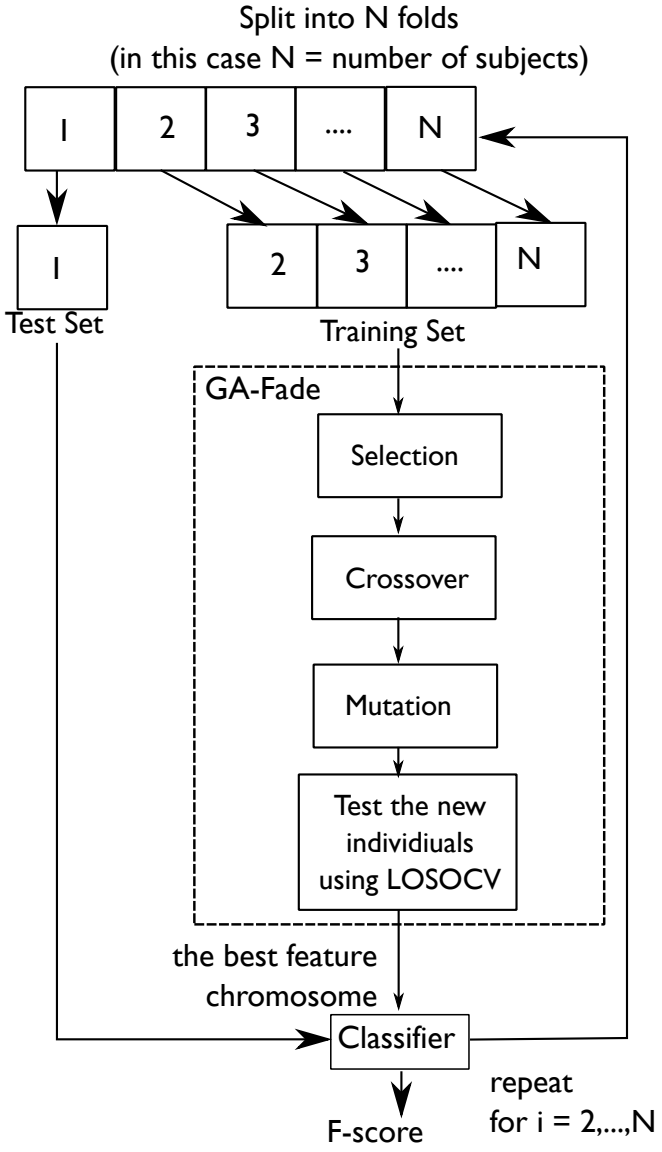


Figure 6.5: GA-Fade validation using LOSOCV

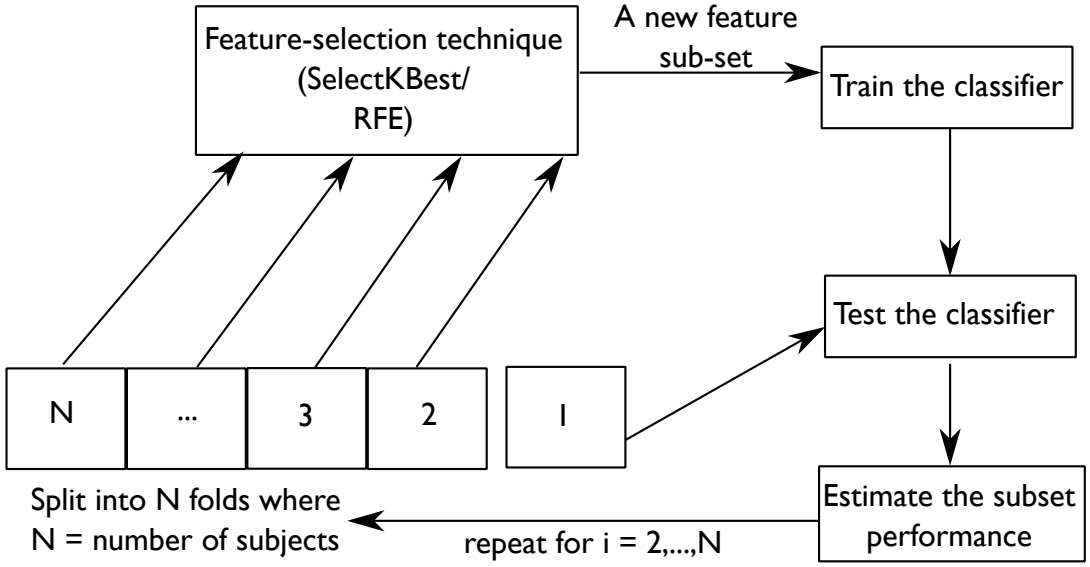


Figure 6.6: Feature-selection techniques (SelectKBest and RFE) validation using LOSOCV

6.3.3 Recursive Feature Elimination (RFE)

Recursive Feature Elimination (RFE) was proposed by Guyon *et al.* [73]. This technique consists of three main steps:

- Train a classifier.
- Compute the ranking (r) of each feature based on its coefficient (w) in the decision function.
- Remove the feature with the lowest rank.

Those steps above are iteratively done until the desired number of features is achieved. Algorithm 6.5 shows the feature-selection process using RFE. In this study, an RFE library from Scikit-learn was used. With regard to the classifier, Logistic Regression (LR) with $C = 10^9$ was used to train the classifier.

Algorithm 6.5 Recursive feature elimination (RFE)

1: $X = [x_1, x_2, x_3, ..., x_n]$ ▷ Training samples
2: $Y = [y_1, y_2, y_3, ..., y_n]$ ▷ Class labels
3: $F = [f_1, f_2, f_3, ..., f_5]$ ▷ Features
4: **while** length of $F >$ desired number of features **do**
5: $w =$ classifier.train(X, Y) ▷ w= features' coefficients in the decision function
6: $r_i = (w_i)^2$ ▷ r_i = rank of the i-th feature
7: $F =$ sort(F) ▷ non-increasingly sorted
8: $F = F[1 : \text{length}(F) - 1]$ ▷ remove the feature with the lowest rank
9: **end while**
10: Return F

6.4 Results and Analysis

6.4.1 GA-Fade results

Figure 6.7 shows the distribution of fitness values produced by GA-Fade for each number of generations. From this figure, it can be seen that increasing the number of generation can increase the fitness value. Based on the Wilcoxon test, increasing the number of generations can significantly increase the fitness value, with p-values ≤ 0.05. Having a high fitness value can lead to the optimal solution. Increasing the size of the initial population does not give a significant improvement in the fitness value (p-values ≥ 0.4). These results show that increasing the number of generations is more important than increasing the initial population to improve the classifier detection rate (precision, recall, and F-score) and reduce the feature-extraction computational cost.

In terms of precision, increasing the number of generation does not significantly affect the classifier performance (p-values≥ 0.1), except when the initial population is 80 and the number of generations is increased from 100 to 200 (p-value $= 4 \times$

Table 6.2: Average and standard deviation of precision, recall, and F-Score of GA-based feature selection with different initial population sizes and different numbers of generations

(a) Precision (%)

Initial population	**Number of generations**		
	50	**100**	**200**
40	96.4±7.9	99±2.3	98.4±3.9
60	98.1±6.5	96.6±6.2	96.8±7.6
80	97.5±7.9	97.3±5.3	96.9±5.1

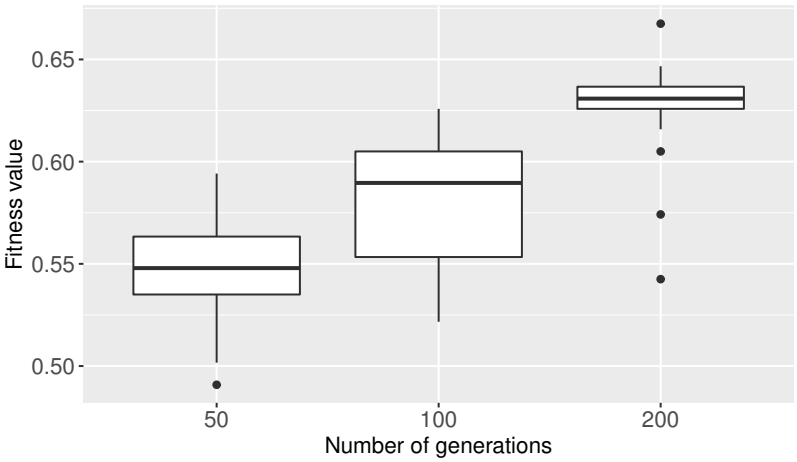
(b) Recall (%)

Initial population	**Number of generations**		
	50	**100**	**200**
40	97.8±3.7	96.6±6.3	97.2±5.5
60	97.1±6.6	97.3±6.1	96.9±5.8
80	97±5.4	96.1±6.6	97.6±5.4

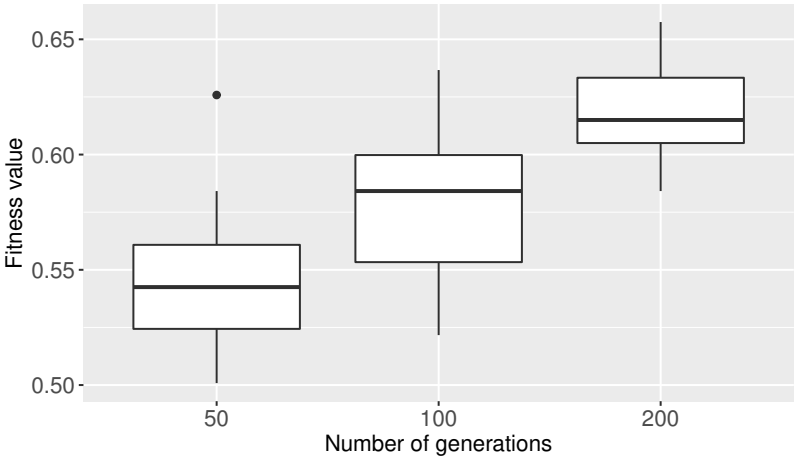
(c) F-score (%)

Initial population	**Number of generations**		
	50	**100**	**200**
40	96.9±4.8	97.7±3.4	97.7±3.9
60	97.4±4.9	96.8±5.1	96.7±5.5
80	97±5.5	96.6±5.3	97.1±4.2

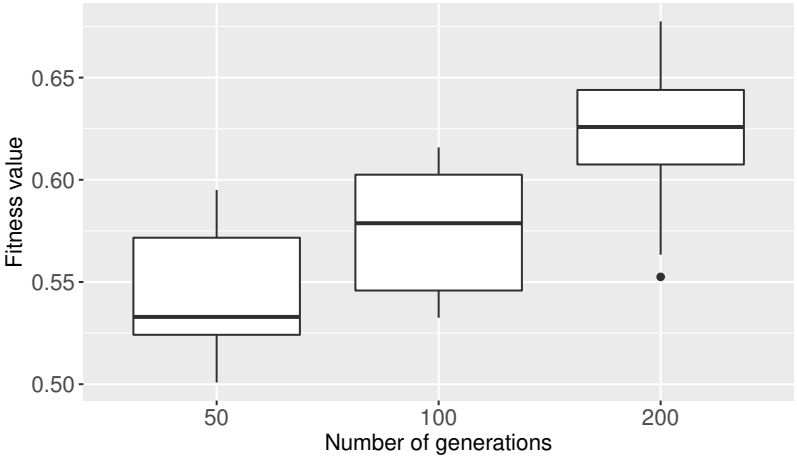
10^{-3}). The recall of the classifier is not significantly affected by the number of generations (p-values ≥ 0.05). The performance of the classifier (in terms of F-score) can be significantly improved when the initial population is 80 and the number of generations is increased from 100 to 200. Increasing the number of generations can significantly affect the computational cost of feature extraction (p-values ≤ 0.02), except when the initial population is 60 and number of generation is increased from 50 to 100 (p-value = 0.08).



(a) Initial population: 40



(b) Initial population: 60



(c) Initial population: 80

Figure 6.7: Fitness-value distributions of GA for several initial population values

Table 6.3: Average and standard deviation of minimum running time (ms) for selected features of GA-based feature selection with different initial population sizes and different numbers of generations

(a) Chest

Initial population	**Number of generations**		
	50	**100**	**200**
40	0.3±0.1	0.2±0.1	0.2±0.1
60	0.3±0.1	0.2±0.1	0.2±0.1
80	0.3±0.1	0.2±0.1	0.2±0.1

(b) Waist

Initial population	**Number of generations**		
	50	**100**	**200**
40	0.3±0.1	0.2±0.1	0.2±0.1
60	0.3±0.1	0.2±0.1	0.2±0.1
80	0.3±0.1	0.2±0.1	0.2±0.1

(c) Thigh

Initial population	**Number of generations**		
	50	**100**	**200**
40	0.3±0.1	0.2±0.1	0.2±0.1
60	0.3±0.1	0.2±0.1	0.2±0.1
80	0.3±0.1	0.2±0.1	0.2±0.1

6.4.2 Performance comparison

6.4.3 Classifier performance (precision, recall, and F-score)

Table 6.4 shows the average of the precision, recall, and F-score of the subset of features that are selected by each feature-selection technique and by using all features. Compared to using all features, using features that are selected by GA-Fade does not give a significant improvement in terms of precision (p-values ≥ 0.2), recall (p-values ≥ 0.3), and F-score (p-values ≥ 0.2). Using either a filter-based or an embedded technique also does not give a significant improvement in terms of precision (p-values ≥ 0.6), recall (p-values ≥ 0.3), and F-score (p-values ≥ 0.3). The experimental results also show that features that are selected using GA-Fade, SelectKBest, or RFE give similar results in terms of precision (p-values ≥ 0.06), recall (p-values ≥ 0.01), and F-score (p-values ≥ 0.08). These results show that using wrapper-based, filter-based, or embedded techniques can give comparable results. Also, reducing the number of features in this study does not significantly improve the performance.

6.4.4 Features computational cost comparison

Table 6.5 shows the computational cost of selected features by each feature-selection technique on each sensor placement. For the chest sensor placement, GA-Fade is able to select features that give a significantly lower computational cost than features that are selected by other feature-selection techniques (p-values $\leq 1.7 \times 10^{-4}$). GA-Fade is also able to select features those can give a significantly less computational cost than features that are selected by SelectKBest and RFE from both waist and thigh sensors (p-values$\leq 1.8 \times 10^{-4}$), except when SelectKBest with f_classif score function is used for the thigh placement. These results show that GA-Fade is able to select features that have a significantly lower computational cost than features that are selected by other feature-selection techniques, in most of the cases. This

Table 6.4: Classifier performance (average and standard deviation) on selected features

Feature-selection technique	**Precision (%)**	**Recall (%)**	**F-score (%)**
Full features	97.3±6.2	96.4±7.6	96.6±5.2
SelectKBest + chi2	96.3±9.1	96±8.4	95.7±5
SelectKBest + f_classif	96.6±9.4	97±4.4	96.5±6.5
RFE+Logistic Regression-based classifier	96.9±6.8	98±3.7	97.3±4

becomes an advantage of GA-Fade compared to other feature-selection techniques, since computational cost is critical for wearable-sensor-based applications, since a wearable device has limited resources (e.g. battery power, memory, and CPU).

6.5 Discussion and limitations

6.5.1 Discussion

6.5.1.1 Performance comparison

This chapter aims to investigate the performance of a proposed genetic-algorithm-based feature-selection technique in selecting features that can give an equal or better detection rate (precision, recall, and F-score) and have less computational cost. The idea of considering both detection rate and computational cost is discussed in Wang *et al.* [154]. Their study uses a wrapper-based feature-selection technique to select features. The first problem of their approach is that they do not provide any

Table 6.5: Computational cost (average and standard deviation) for selected features for each placement

Feature selection technique	**Chest (ms)**	**Waist (ms)**	**Thigh (ms)**
Full features	0.8±0.0	0.8±0.0	0.8±0.0
SelectKBest + chi2	0.3±0.0	0.3±0.0	0.4±0.0
SelectKBest + f_classif	0.5±0.0	0.3±0.0	0.1±0.0
RFE+Logistic Regression-based classifier	0.7±0.1	0.5±0.1	0.4±0.1

justification of a feature removal in each iteration. For example, they do not provide any explanation regarding the removal of f_{10} in the second iteration (see Table VI from their paper). In GA-Fade, the removal of features is done through crossover and mutation processes from the selected subsets. The second problem of Wang *et al.*'s approach is that it does not select the best feature subset among the evaluated feature subsets, while GA-Fade always chooses the best subset (chromosome) among the evaluated subsets. Another issue of Wang *et al.*'s study is that they do not compare their approach with other feature selection techniques. Thus, it is not clear whether it is worthwhile to implement a wrapper-based feature-selection technique in this case (note that, since the wrapper-based feature-selection technique has a more complex process, it takes more time to select features than using a filter-based or an embedded technique).

This study compares GA-Fade with other feature-selection techniques from each category (SelectKBest is a filter-based technique and RFE is an embedded technique). The results of this comparison show that GA-Fade, SelectKBest, RFE have an equal capability of choosing features that can give a detection rate (precision, recall, and F-score) equal to the condition when all features are used. However, Tables 6.3 and 6.5 show that GA-Fade is able to select features that have a sig-

nificantly lower computational cost than features that are selected by SelectKBest and RFE. This causes GA-Fade to become superior to SelectKBest and RFE. A lesson learned from this chapter is that the wrapper-based feature-selection technique is preferred when the selection process involves multiple criteria (detection rate, computational cost, number of sensors). This study also shows that a heuristic search approach such as a genetic algorithm can be used to select features to get an optimum detection rate with less computational cost.

6.5.1.2 Selected features and a risk of overfitting

Since each validation is done using inner and outer cross-validation (see Figure 6.5), each iteration produces a different set of sub-features. Figure 6.8 shows the frequency of each feature on each placement, while Table 6.7 shows the 12 most-picked features from each sensor placement. Table 6.7 shows that using domain knowledge in selecting features is less useful than using a feature-selection technique in this case. For example, studies from Bourke *et al.* [25], Dumitrache *et al.* [47], and Sorvala *et al.* [136] use maximum acceleration during the impact stage of fall as a feature to classify falls from other activities. Note that these studies place their sensor on the waist of the subject. Based on the results of Table 6.7, the maximum acceleration during the impact stage is not included in the twelve-most-picked features (see the waist column). Thus, selecting features using a feature-selection technique performs better than selecting features manually using domain knowledge in this case.

Since the F-score of using all features and selected features (regardless of the feature selection technique) are similar, this shows that there are redundant features used in this thesis. Thus, doing a feature selection, in this case, is necessary to reduce the computational cost of the system. Another issue that appears in this thesis is a risk of overfitting because the number of features is high. Since this chapter uses only the Cogent dataset, the risk of overfitting in this chapter is reduced by implementing the inner and outer cross-validation [116]. Another way to reduce

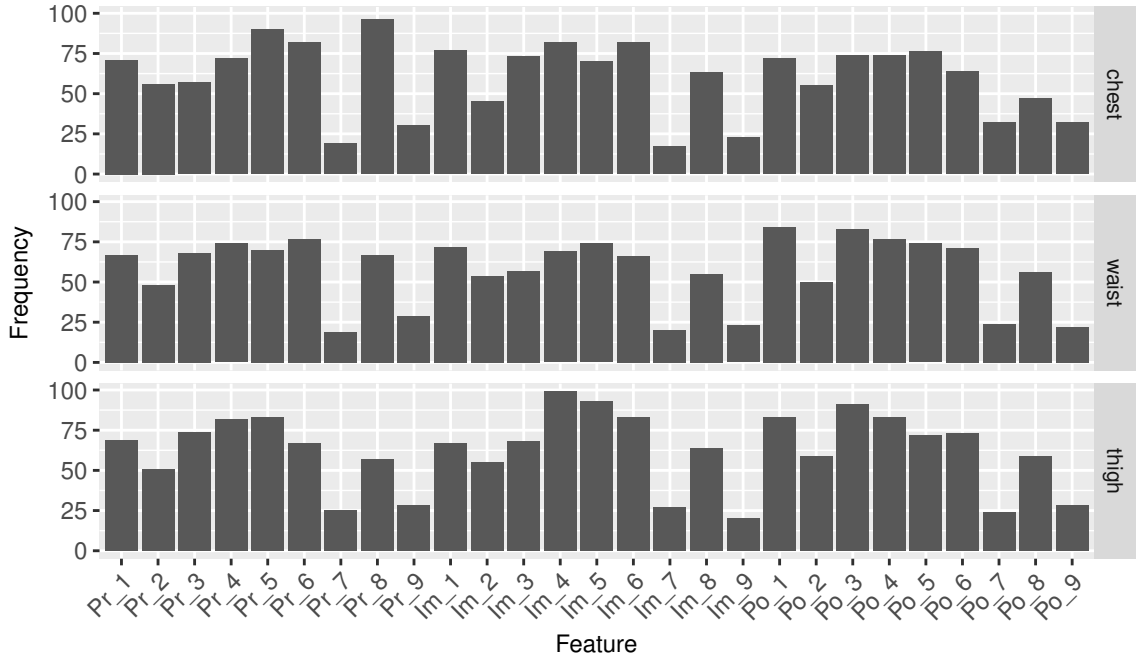


Figure 6.8: Frequency of each feature being chosen by GA-Fade. Pr, Im, and Po mean pre-impact, impact, and post-impact, respectively, and the number represents the feature index. Table 6.6 shows the features that are used for this study together with their indices.

the risk of overfitting is by tuning the hyper-parameters of the machine-learning algorithm [125]. However, this is out of the scope of this thesis and is left for future work.

6.5.2 Limitations

6.5.2.1 Data pre-processing and the machine-learning algorithm choice

The first limitation of this study is the choice of the machine-learning algorithm and features. The results of this study might change when the feature or the machine-learning algorithm is changed. Because a data-imputation process is implemented before the feature-selection process, the results provided in this chapter may be different from the results from the real implementation. However, by comparing the results of this chapter with the results from Table 5.16, the difference should not be expected to be significant. To avoid the data-imputation process, a synchronisation

Table 6.6: Index of the used features

Feature name	Index
Mean acceleration (Mean)	1
Variance of acceleration (Variance)	2
Maximum acceleration (Max)	3
Minimum acceleration (Min)	4
Root-mean-square acceleration (RMS)	5
Velocity	6
Signal-magnitude area (SMA)	7
Exponential moving average (EMA)	8
Energy	9

process is needed. Data synchronisation is one of the important problems in the data pre-processing stage, especially for sensor-based healthcare technology [137], and some studies proposed a time-based synchronisation approach to synchronise data from multiple sensors [17, 49, 59]. This synchronisation issue is left for future work. Also, because this chapter uses data from the Cogent dataset only, the results cannot be generalised. Thus, more data (especially data from older subjects) are needed for the evaluation.

6.5.2.2 A multi-sensor system for fall detection

Lara and Labrador [98] suggest using as few sensors as possible to increase the user comfort and to reduce the complexity and energy consumption of the system since fewer data are processed. However, this study shows that all the feature-selection techniques (GA-Fade, SelectKBest, RFE) choose to select features from all sensors. This means that better features are needed to get a better detection rate with fewer sensors. Also, an in-garment technology (e.g. Jung *et al.* [83]) can be implemented to get the benefits of the multi-sensor-based system without sacrificing user's comfort.

Table 6.7: The 12 most-picked features by GA-Fade from each sensor placement

Chest	**Waist**	**Thigh**
Pre-impact : EMA (Pr_8)	Post-impact : Mean (Po_1)	Impact : Min (Im_4)
Pre-impact : RMS (Pr_5)	Post-impact : Max (Po_3)	Impact : RMS (Im_5)
Pre-impact : Velocity (Pr_6)	Pre-impact : Velocity (Pr_6)	Post-impact : Max (Po_3)
Impact : Min (Im_4)	Post-impact : Min (Po_4)	Pre-impact : RMS (Pr_5)
Impact : Velocity (Im_6)	Pre-impact : Min (Pr_4)	Impact : Velocity (Im_6)
Impact : Mean (Im_1)	Impact : RMS (Im_5)	Post-impact : Mean (Po_1)
Post-impact : RMS (Po_5)	Post-impact : RMS (Po_5)	Post-impact : Min (Po_4)
Post-Impact : Max (Po_3)	Impact : Mean (Im_1)	Pre-impact : Min (Pr_4)
Post-impact : Min (Po_4)	Post-impact : Velocity (Po_6)	Pre-impact : Max (Pr_3)
Impact : Max (Im_3)	Pre-impact : RMS (Pr_5)	Post-impact : Velocity (Po_6)
Pre-impact : Min (Pr_4)	Impact : Min (Im_4)	Post-impact : RMS (Po_5)
Post-impact : EMA (Po_1)	Pre-impact : Max (Pr_3)	Pre-impact : Mean (Pr_1)

6.5.2.3 Power efficient design

The energy consumption of the real implementation on the real device can be very different. This is because the highest energy consumption comes from the radio transmission module [154], where this energy consumption is assumed to be zero in this thesis. Since this thesis uses the runtime to represent the computational-cost without considering other possible aspects (for example the radio transmission), this is very limited to the computational cost of the feature extraction.

6.6 Chapter summary

This chapter provides a genetic-algorithm-based feature-selection technique (GA-Fade) for fall detection using wearable sensors. This technique tackles the multi-criteria issue, which considers F-score, computational cost, and number of sensors as the selection criteria. GA-Fade has an ability to select a subset of features from three different sensor placements, which can give a significantly better F-score, while having a relatively low computational cost.

GA-Fade is able to select features that can reduce, by around 60%, the total number of features, and achieves a precision of up to 99%, a recall of up to 97.8%, and an F-score of up to 97.7% on average. This result is equal to results from features that are selected by the SelectKBest and RFE techniques. Regarding the computational cost, features that are selected by GA-Fade have the significantly lowest total computational cost among the techniques. This is an advantage of GA-Fade.

In response to **RQ-4,** using a genetic algorithm with a penalty-based fitness function is able to select a subset of features that have comparable precision, recall, and F-score but significantly lower computational cost than with all features and features that are selected by both the SelectKBest and RFE techniques, where these techniques use a single criterion (detection rate). Because GA-Fade adopts

the wrapper-based feature-selection technique, the results shown in this chapter indicate that the wrapper-based feature-selection technique performs better than both filter-based and embedded feature-selection techniques, when multiple criteria are used (detection rate, computational cost, and number of sensors).

Chapter 7

Conclusions and Future Work

7.1 Conclusion

Having a reliable fall-detection system for older people is highly desirable, as it can reduce the complications that might be produced by unnoticed falls. As the size of wearable sensors is getting smaller and their price is getting lower, these technologies become convenient to be used for fall detection. Key research in fall detection using wearable sensors is developing a fall-detection approach that can give a high detection rate, while reducing the system's computational cost. Having a less-computational-cost approach is an advantage, as the wearable sensors have limited resources such as memory and battery capacity. This thesis aimed to investigate and develop a machine-learning-based approach that is suitable for wearable-sensor-based fall detection. To ensure the reproducibility and the fairness of the results, publicly accessible datasets: Cogent, SisFall, and FARSEEING, were used in this thesis. A leave-one-subject-out cross-validation (LOSOCV) is mainly used to evaluate the performance of the classifier, since for fall detection, the main source of variation is due to characteristics of the subjects or how sensors are attached rather than, say, the time of day or the temperature in the room.

This thesis has addressed the issue of both the detection rate and the computational cost, and it provides the following contributions:

1. A study of both threshold-based and machine-learning-based fall-detection approaches using publicly accessible datasets. This study aims to analyse the use of the sliding-window technique for data segmentation on the machine-learning-based approach. The experiments show that using a larger Fixed-size Non-Overlapping Sliding Window (FNSW) does not necessarily increase the classifier's precision, recall, and F-score. Moreover, using a larger window overlap for a fixed-size overlapping sliding window (FOSW) can increase the number of false alarms (reduction in precision). Also, a fair comparison has been done in this study to analyse whether the sliding-window-based machine-learning approach can perform better than a threshold-based approach. The experiment shows that the sliding-window-based machine-learning approach is able to outperform the threshold-based approach, though the differences are not significant when the Cogent dataset is used, regardless of the sliding window technique. The machine-learning-based approach can achieve an F-score of up to 96.5%, whereas the threshold-based approach can only achieve up to an 88.6% F-score.

2. An event-triggered machine-learning approach (**EvenT-ML**), where this approach extracts features based on the state of the body (active or inactive) and fall stages (pre-impact, impact, and post-impact) aiming to increase the performance of the classifier. This approach achieves a significantly better performance than the sliding-window-based (Fixed-size Non-overlapping Sliding Window (FNSW) and Fixed-size Overlapping Sliding Window (FOSW)) approach, an existing fall-stage-based [127] approach, and an existing threshold-based approach (IMPACT+POSTURE [86]) with an F-score of up to 97.6%. Also, as an additional advantage, Event-ML has a significantly lower computational cost than both FNSW- and FOSW-based machine-learning approaches.

3. A genetic-algorithm-based feature-selection technique for fall detection (**GA-Fade**), to select a subset of features based on the detection rate (F-score), computational cost (running time), and number of sensors being used. Compared to features that are selected by filter (SelectKBest [125]) and embedded (Recursive Feature Elimination (RFE) [73]) feature-selection techniques, where these techniques are examples of single-criterion-based feature-selection techniques, GA-Fade can select features from three different placements of sensors that are able to give a comparable F-score (with an F-score of up to 97.7%) and a significantly lower total computational cost in most of the cases.

The next section provides answers to the research questions posed in Chapter 1.

7.2 Answers to research questions

This thesis has examined the following research questions:

RQ1: What is the impact of the sliding-window type and size on the classifier detection rate (in terms of precision, recall, and F-score) when the machine-learning based approach is used?

For the FNSW-based machine-learning approach, using a larger FNSW does not necessarily increase the performance of the classifier in terms of precision, recall, and F-score, unless the length of the activity is uniform. A larger window size (15 seconds) is suitable for the SisFall dataset as it has a uniform length of fall, while 2 seconds of FNSW is suitable for the Cogent dataset which has a more varied length of fall. With the FOSW-based machine-learning approach, increasing the window-overlap size can cause an increase in false alarms (decrease in precision) in most cases. This is because the number of data overlaps between fall and non-fall data is increased when the window overlap size is increased. Another important finding from this investigation is that there is a relatively big gap between precision and

recall, which makes the FNSW- and FOSW-based machine-learning approaches still not applicable for real-world situations.

RQ2: Does the sliding-window machine-learning based approach provide a significantly better detection rate than the threshold-based approach on publicly accessible datasets?

By using Logistic Regression (LR) with an inverse of regularisation strength (C) equal to 10^9 and a 2-second Fixed-size Non-overlapping Sliding Window (FNSW), the machine-learning based approach is able to outperform the threshold-based approach in terms of F-score using the Cogent dataset. By using an SVM algorithm with a radial-basis-function (RBF) kernel, a 2-second Fixed-size Overlapping Sliding Window (FOSW), and a 90% data overlap, the machine-learning based approach is able to outperform the threshold-based approach in terms of F-score. However, both FNSW- and FOSW-based approaches are unable to achieve significantly different results using the Cogent dataset. On the other hand, for the SisFall dataset, by using a k-NN-based classifier (with k=2), both FNSW- and FOSW-based machine learning approaches are able to significantly outperform IMPACT+POSTURE in terms of F-Score.

Overall, the machine-learning-based approach can provide a better performance than the threshold-based approach. But the difference is not significant when the Cogent dataset is used, regardless of the sliding-window technique. A more detailed investigation about this comparison is provided in Section 4.5.

RQ3: Does correctly aligning a segment with the fall stages (pre-impact, impact, and post-impact) and using the state of the body of the subject (active or inactive) to trigger the feature-extraction and classification processes improve both the system's detection rate and reduce its computational cost?

Yes.

Correctly aligning a segment of accelerometer signal with fall stages before doing feature extraction can significantly improve the classifier's detection rate (F-score), while reducing its computational cost. In order to answer this question, a novel event-triggered machine-learning approach (EvenT-ML) was developed. EvenT-ML correctly aligns a segment with fall stages by resolving the multi-peak issue (Figure 2.5). To reduce the computational cost of the system, EvenT-ML has an ability to prevent the feature extraction from being executed for all possible segments, by triggering the feature-extraction and classification processes only when the state of the subject is active.

Compared to the FNSW- and FOSW-based machine-learning approaches, EvenT-ML can achieve a significantly better precision and F-score, while still maintaining a relatively good recall on both datasets. EvenT-ML is able to achieve up to 97.6% (Cogent dataset) and 91.3% (SisFall dataset) F-scores. In terms of computational cost, EvenT-ML is able to achieve a reduction by a factor of 8 (on average) compared to FNSW and of 80 (on average) compared to FOSW for the Cogent dataset. For the SisFall dataset, a reduction by factor of 2 compared to FNSW can be achieved when EvenT-ML is used. Compared to FOSW on the SisFall dataset, EvenT-ML is able to achieve up to a factor of 20 reduction. EvenT-ML is also able to reduce the gap between precision and recall, making this approach more applicable to real-world cases than both the FNSW- and FOSW-based machine-learning approaches.

Compared to the threshold-based approach, EvenT-ML is able to achieve a better precision, a better recall, and a better F-score in most cases. In general, EvenT-ML

is able to significantly outperform IMPACT+POSTURE on both datasets regardless of the machine-learning algorithm, in terms of F-score. Regarding the sensor placement, chest placement produces the best results for EvenT-ML, followed by the waist and thigh placements. This study also found that the Cogent dataset is better used to train the classifier than is the SisFall dataset. This is because using the Cogent dataset (evaluated using the hold-out technique) can give similar results to using the FARSEEING dataset (evaluated using leave-one-subject-out cross-validation), whereas using the SisFall dataset can give a significantly lower performance (in terms of F-score) when CART or LR is used to train the classifier. This finding also confirmed that using the Cogent dataset, where all the subjects are young and healthy, can give comparable results with using real-fall data from older people. Chapter 5 provides a detailed investigation of EvenT-ML and its performance.

RQ4: Does a meta-heuristic search technique (genetic algorithm) select features that have a higher detection rate and a lower computational cost than features that are selected by single-criterion-based feature-selection techniques (filter-based and embedded techniques)?

Using a genetic algorithm (GA-Fade) to select a subset of features using a penalty-based function can select features that have a comparable F-score, with a significantly lower computational cost than features that are selected by SelectKBest (filter-based technique) and Recursive Feature Elimination (embedded feature-selection technique). These results confirm that the wrapper-based feature-selection technique (with a multi-criteria-based fitness function) is better than both filter-based and embedded techniques for fall detection using wearable sensors. For its penalty function, F-score (detection rate), running time (computational cost), and number of sensors used are implemented. With a Logistic Regression (LR)-based classifier, an F-Score of 97.7% (on average) can be achieved by a subset of features that are selected by GA-Fade.

7.3 Future Work

This thesis has successfully met the research aims proposed in Section 1.2, by answering all the research questions. However, there are several areas of future work that can be investigated as an expansion of the work presented in this thesis. This section provides several areas that can be investigated to extend the scope of this study.

7.3.1 An implementation of EvenT-ML on a real device

This thesis tries to develop an automated fall-detection and alerting system using sensors that are attached to clothes. All experiments conducted in this study are based on simulations, which were all done on a PC. Thus, the impact of EvenT-ML on battery life has not been explicitly evaluated. Therefore, future work will investigate the energy consumption of EvenT-ML on a real wearable device.

7.3.2 An investigation on more possible features and an improvement of the machine-learning algorithm

This study uses 9 types of features: minimum, maximum, average, variance, root mean square of acceleration vector magnitude, velocity, energy, acceleration exponential moving average, and signal-magnitude area. Gonzáles *et al.* [68] summarise features for human-activity recognition. Some features from Gonzáles *et al.*'s study that could be investigated to improve the classifier detection rate are:

- Tilt of the body : $\frac{1}{w}\sum_{t=i}^{i+w} \mid a_i^y \mid + \mid a_i^z \mid$,
- The intensity of the movement (InMo): $\frac{1}{w}\sum_{i=1}^{w} \mid \frac{a_{t-i}^{\{x,y,z\}} - a_{t-i-1}^{\{x,y,z\}}}{\Delta x_t} \mid$,
- Time between peaks,

where a, w, and Δx_t are the acceleration signal, number of samples in a segment, and a representation of the time between samples (this can be ignored if the sampling rate is constant). The next improvement that can be made is increasing the detection rate by tuning the hyper-parameters of the machine-learning algorithms.

7.4 Summary

In conclusion, this thesis has demonstrated that the detection rate of a fall-detection system (in terms of precision, recall, and F-score) can be significantly increased, while its computational cost can be reduced. An event-triggered machine-learning based fall-detection approach (EvenT-ML) has been proposed, where this technique is able to segment features and correctly align fall stages (pre-impact, impact, and post-impact). Then those stages are used as a basis for feature extraction. Three publicly accessible datasets have been used, so that the comparison can be fairly done. Also, this study shows that using data from young subjects (the Cogent dataset) to train the classifier is as effective as using data from older people (the FARSEEING dataset). To increase the detection rate of the system, a feature reduction has to be done to eliminate a number of redundant features. Also, reducing the number of features can reduce the computational cost of the system. This study proposes a genetic-algorithm-based feature selection (GA-Fade). This technique is able to select a significantly better sub-set of features, with a lower computational cost, from various sensor placements.

References

[1] The farseeing real-world repository: a large-scale collaborative database to collect and share sensors signals from real-world falls. http://farseeingresearch.eu/the-farseeing-real-world-fall-repository-a-large-scale-collaborative-database-to-collect-and-share-sensor-signals-from-real-world-falls/. Accessed: 17 October 2017.

[2] Stefano Abbate, Marco Avvenuti, Francesco Bonatesta, Guglielmo Cola, Paolo Corsini, and Alessio Vecchio. A smartphone-based fall detection system. *Pervasive and Mobile Computing*, 8(6):883–899, 2012.

[3] Age UK. Stop Falling : Start Saving Lives and Money. pages 1–23, 2013.

[4] Syed Farooq Ali, Reamsha Khan, Arif Mahmood, Malik Tahir Hassan, and Moongu Jeon. Using temporal covariance of motion and geometric features via boosting for human fall detection. *Sensors*, 18(6), 2018.

[5] Atika Arshad, Sheroz Khan, A. H. M. Z. Alam, Ahmad F. Ismail, and Rumana Tasnim. Capacitive proximity floor sensing system for elderly tracking and fall detection. In *IEEE 4th International Conference on Smart Instrumentation, Measurement and Application (ICSIMA)*, pages 1–5, November 2017.

[6] Danielsen Asbjørn and Torresen Jim. Recognizing bedside events using thermal and ultrasonic readings. *Sensors*, 17(6), 2017.

[7] Louis Atallah, Benny Lo, Rachel King, and Guang-Zhong Yang. Sensor positioning for activity recognition using wearable accelerometers. *IEEE Transactions on Biomedical Circuits and Systems*, 5(4):320–329, August 2011.

[8] Omar Aziz, Magnus Musngi, Edward J. Park, Greg Mori, and Stephen N. Robinovitch. A comparison of accuracy of fall detection algorithms (threshold-based vs. machine learning) using waist-mounted tri-axial accelerometer signals from a comprehensive set of falls and non-fall trials. *Medical & Biological Engineering & Computing*, 55(1):45–55, January 2017.

[9] Omar Aziz, Colin M. Russell, Edward J. Park, and Stephen N. Robinovitch. The effect of window size and lead time on pre-impact fall detection accuracy using support vector machine analysis of waist mounted inertial sensor data. In *36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 30–33, Aug 2014.

[10] Fabio Bagalà, Clemens Becker, Angelo Cappello, Lorenzo Chiari, Kamiar Aminian, Jeffrey M. Hausdorff, Wiebren Zijlstra, and Jochen Klenk. Evaluation of accelerometer-based fall detection algorithms on real-world falls. *PLoS ONE*, 7(5):1–9, 2012.

[11] Ying-Wen Bai, Shiao-Chian Wu, and Chia-Hao Yu. Recognition of direction of fall by smartphone. In *26th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1–6, May 2013.

[12] Tanvi Banerjee, James M. Keller, Marjorie Skubic, and Erik Stone. Day or night activity recognition from video using fuzzy clustering techniques. *IEEE Transactions on Fuzzy Systems*, 22(3):483–493, June 2014.

[13] Oresti Banos, Juan-Manuel Galvez, Miguel Damas, Hector Pomares, and Ignacio Rojas. Window size impact in human activity recognition. *Sensors (Basel, Switzerland)*, 14(4):6474–6499, 2014.

[14] Angela Barriga, José M. Conejero, Juan Hernández, Elena Jurado, Enrique Moguel, and Fernando Sánchez-Figueroa. A vision-based approach for building telecare and telerehabilitation services. *Sensors*, 16(10), 2016.

[15] C. Becker, L. Schwickert, S. Mellone, F. Bagalà, L. Chiari, J.L. Helbostad, W. Zijlstra, K. Aminian, A. Bourke, C. Todd, S. Bandinelli, N. Kerse, and J. Klenk. Proposal for a multiphase fall model based on real-world fall recordings with body-fixed sensors. *Zeitschrift für Gerontologie und Geriatrie*, 45(8):707–715, December 2012.

[16] Yoshua Bengio and Yves Grandvalet. No unbiased estimator of the variance of k-fold cross-validation. *J. Mach. Learn. Res.*, 5:1089–1105, December 2004.

[17] Terrell R. Bennett, Nicholas Gans, and Roozbeh Jafari. Multi-sensor data-driven: Synchronization using wearable sensors. In *Proceedings of the 2015 ACM International Symposium on Wearable Computers*, ISWC '15, pages 113–116, New York, NY, USA, 2015. ACM.

[18] Richard A. Berk. *Classification and Regression Trees (CART)*, pages 1–65. Springer New York, New York, NY, 2008.

[19] Sebastian D. Bersch, Djamel Azzi, Rinat Khusainov, Ifeyinwa E. Achumba, and Jana Ries. Sensor data acquisition and processing parameters for human activity classification. *Sensors (Basel, Switzerland)*, 14(3):4239–4270, 2014.

[20] Sourav Bhattacharya, Petteri Nurmi, Nils Hammerla, and Thomas Plötz. Using unlabeled data in a sparse-coding framework for human activity recognition. *Pervasive and Mobile Computing*, 15:242–262, 2014. Special Issue on Information Management in Mobile Applications, Special Issue on Data Mining in Pervasive Environments.

[21] Jerrel Blankenship, Matthew Bussa, and Scott Millett. *eXtreme Programming*, pages 29–51. Apress, Berkeley, CA, 2011.

[22] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A Training Algorithm for Optimal Margin Classifiers. *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152, 1992.

[23] Alan K. Bourke, Jochen Klenk, Lars Schwickert, Kamiar Aminian, Espen A. F. Ihlen, Sabato Mellone, Jorunn L. Helbostad, Lorenzo Chiari, and Clemens Becker. Fall detection algorithms for real-world falls harvested from lumbar sensors in the elderly population: A machine learning approach. In *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 3712–3715, Aug 2016.

[24] Alan K. Bourke, Karol J. O'Donovan, John Nelson, and Gearoid M. OLaighin. Fall-detection through vertical velocity thresholding using a tri-axial accelerometer characterized using an optical motion-capture system. In *30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 2832–2835, Vancouver, BC, August 2008.

[25] Alan K. Bourke, Pepijn van de Ven, Mary Gamble, Raymond O'Connor, Kieran Murphy, Elizabeth Bogan, Eamonn McQuade, Paul Finucane, Gearóid ÓLaighin, and John Nelson. Assessment of waist-worn tri-axial accelerometer based fall-detection algorithms using continuous unsupervised activities. In *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 2782–2785, August 2010.

[26] Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and Regression Trees*. Wadsworth International Group, Belmont, California, 1984.

[27] Simon Brownsell and Mark S Hawley. Automatic fall detectors and the fear of falling. *Journal of Telemedicine and Telecare*, 10(5):262–266, 2004.

[28] James Brusey, John Kemp, Elena Gaura, Ross Wilkins, and Mike Allen. En-

ergy profiling in practical sensor networks: Identifying hidden consumers. *IEEE Sensors Journal*, 16(15):6072–6080, August 2016.

[29] James Brusey, Ramona Rednic, Elena I. Gaura, John Kemp, and Nigel Poole. Postural activity monitoring for increasing safety in bomb disposal missions. *Measurement Science and Technology*, 20(7):075204, 2009.

[30] Andreas Bulling, Ulf Blanke, and Bernt Schiele. A tutorial on human activity recognition using body-worn inertial sensors. *ACM Computing Survey*, 46(3):33:1–33:33, January 2014.

[31] Adrian Burns, Barry R. Greene, Michael J. McGrath, Terrance J. O'Shea, Benjamin Kuris, Steven M. Ayer, Florin Stroiescu, and Victor Cionca. SHIMMER–A Wireless Sensor Platform for Noninvasive Biomedical Research. *IEEE Sensors Journal*, 10(9):1527–1534, 2010.

[32] Eduardo Casilari, Jose A. Santoyo-Ramón, and Jose M. Cano-García. Umafall: A multisensor dataset for the research on automatic fall detection. *Procedia Computer Science*, 110:32–39, 2017. 14th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2017) / 12th International Conference on Future Networks and Communications (FNC 2017) / Affiliated Workshops.

[33] Eduardo Casilari, José-Antonio Santoyo-Ramón, and José-Manuel Cano-García. Analysis of public datasets for wearable fall detection systems. *Sensors*, 17(7), 2017.

[34] Jay Chen, Karric Kwong, Dennis Chang, Jerry Luk, and Ruzena Bajcsy. Wearable sensors for reliable fall detection. In *27th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, volume 4, pages 3551–3554, Shanghai, 2005.

[35] Wen-Chang Cheng and Ding-Mao Jhan. Triaxial accelerometer-based fall detection method using a self-constructing cascade-adaboost-svm classifier. *IEEE Journal of Biomedical and Health Informatics*, 17(2):411–419, March 2013.

[36] HyunGi Cho, Suyong Yeon, Hyunga Choi, and Nakju Doh. Detection and compensation of degeneracy cases for imu-kinect integrated continuous slam with plane features. *Sensors*, 18(4), 2018.

[37] Y. Choi, A. S. Ralhan, and S. Ko. A study on machine learning algorithms for fall detection and movement classification. In *International Conference on Information Science and Applications (ICISA)*, pages 1–8, Jeju Island, 2011. IEEE.

[38] Jitender Choudhari and Ugrasen Suman. Extended iterative maintenance life cycle using extreme programming. *SIGSOFT Softw. Eng. Notes*, 39(1):1–12, February 2014.

[39] Michelle Clifford. Human Fall Detection Using 3-Axis Accelerometer. 2005.

[40] Koldo de Miguel, Alberto Brunete, Miguel Hernando, and Ernesto Gambao. Home camera-based fall detection system for the elderly. *Sensors*, 17(12), 2017.

[41] Lariza L. de Oliveira, Alex A. Freitas, and Renato Tinós. Multi-objective genetic algorithms in the study of the genetic codeś adaptability. *Information Sciences*, 425:48–61, 2018.

[42] Janez Demšar. Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[43] Nguyen N. Diep, Cuong Pham, and Tu M. Phuong. A classifier based approach to real-time fall detection using low-cost wearable sensors. In *International*

Conference of Soft Computing and Pattern Recognition (SoCPaR), pages 105–110, Hanoi, December 2013.

[44] Christoph Dinh and Matthias Struck. A new real-time fall detection approach using fuzzy logic and a neural network. In *6th International Workshop on Wearable Micro and Nano Technologies for Personalized Health (pHealth)*, pages 57–60, Oslo, June 2009.

[45] Joanna H. Downton. *Falls in the Elderly*. Edward Arnold London, 1993.

[46] M. K. Dremina, I. Alborova, and L. N. Anishchenko. Importance of the bio-radar signal preprocessing in fall detection. In *Progress In Electromagnetics Research Symposium - Spring (PIERS)*, pages 699–703, May 2017.

[47] Mihail Dumitrache and Sever Pasca. Fall detection algorithm based on triaxial accelerometer data. In *E-Health and Bioengineering Conference (EHB)*, pages 1–4, November 2013.

[48] Adam Dunkels, Fredrik Osterlind, Nicolas Tsiftes, and Zhitao He. Software-based on-line energy estimation for sensor nodes. In *Proceedings of the 4th Workshop on Embedded Networked Sensors*, EmNets '07, pages 28–32, New York, NY, USA, 2007. ACM.

[49] Jeremy Elson, Lewis Girod, and Deborah Estrin. Fine-grained network time synchronization using reference broadcasts. *SIGOPS Oper. Syst. Rev.*, 36(SI):147–163, December 2002.

[50] S. Z. Erdogan and T. T. Bilgin. A data mining approach for fall detection by using k-nearest neighbour algorithm on wireless sensor network data. *IET Communications*, 6(18):3281–3287, 2012.

[51] Baris Erol, Moeness G. Amin, and Boualem Boashash. Range-doppler radar

sensor fusion for fall detection. In *IEEE Radar Conference (RadarConf)*, pages 0819–0824, May 2017.

[52] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[53] Guodong Feng, Jiechao Mai, Zhen Ban, Xuemei Guo, and Guoli Wang. Floor pressure imaging for fall detection with fiber-optic sensors. *IEEE Pervasive Computing*, 15(2):40–47, April 2016.

[54] Andy Field. *Discovering Statistics Using SPSS*. SAGE Publications, London, 2009.

[55] Peter Flach and Meelis Kull. Precision-recall-gain curves: Pr analysis done right. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 838–846. Curran Associates, Inc., 2015.

[56] Jane Fleming and Carol Brayne. Inability to get up after falling, subsequent time on floor, and summoning help: prospective cohort study in people over 90. *BMJ (Clinical research ed.)*, 337:a2227, 2008.

[57] Korbinian Frank, Maria Josefa Vera Nadales, Patrick Robertson, and Tom Pfeifer. Bayesian recognition of motion related activities with inertial sensors. In *Proceedings of the 12th ACM International Conference Adjunct Papers on Ubiquitous Computing - Adjunct*, UbiComp '10 Adjunct, pages 445–446, New York, NY, USA, 2010. ACM.

[58] Alex A. Freitas. A critical review of multi-objective optimization in data mining: A position paper. *SIGKDD Explor. Newsl.*, 6(2):77–86, December 2004.

[59] Saurabh Ganeriwal, Ram Kumar, and Mani B. Srivastava. Timing-sync protocol for sensor networks. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, SenSys '03, pages 138–149, New York, NY, USA, 2003. ACM.

[60] Carmine Garripoli, Marco Mercuri, Peter Karsmakers, Ping J. Soh, Giovanni Crupi, Guy A. E. Vandenbosch, Calogero Pace, Paul Leroux, and Dominique Schreurs. Embedded dsp-based telehealth radar system for remote in-door fall detection. *IEEE Journal of Biomedical and Health Informatics*, 19(1):92–101, January 2015.

[61] Samuele Gasparrini, Enea Cippitelli, Susanna Spinsante, and Ennio Gambi. A depth-based fall detection system using a kinect sensor. *Sensors*, 14(2):2756–2775, 2014.

[62] Elena Gaura, John Kemp, and James Brusey. Leveraging knowledge from physiological data: On-body heat stress risk prediction with sensor networks. *IEEE Transactions on Biomedical Circuits and Systems*, 7(6):861–870, 2013.

[63] Hassan Ghasemzadeh, Navid Amini, Ramyar Saeedi, and Majid Sarrafzadeh. Power-aware computing in wearable sensor networks: An optimal feature selection. *IEEE Transactions on Mobile Computing*, 14(4):800–812, April 2015.

[64] Hristijan Gjoreski, Mitja Lustrek, and Matjaz Gams. Accelerometer placement for posture recognition and fall detection. In *Seventh International Conference on Intelligent Environments*, pages 47–54, July 2011.

[65] Patrice Godefroid and Sarfraz Khurshid. Exploring very large state spaces using genetic algorithms. In Joost-Pieter Katoen and Perdita Stevens, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, pages 266–280, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[66] A. Godfrey, A. K. Bourke, G. M. ÓLaighin, P. van de Ven, and J. Nelson. Activity classification using a single chest mounted tri-axial accelerometer. *Medical Engineering & Physics*, 33(9):1127–1135, 2011.

[67] A. Godfrey, R. Conway, D. Meagher, and G. ÓLaighin. Direct measurement of human movement by accelerometry. *Medical Engineering & Physics*, 30(10):1364–1386, 2008. Special issue to commemorate the 30th anniversary of Medical Engineering & Physics.

[68] Silvia González, Javier Sedano, José R. Villar, Emilio Corchado, Álvaro Herrero, and Bruno Baruque. Features and models for human activity recognition. *Neurocomputing*, 167(C):52–60, November 2015.

[69] Tao Gu, Zhanqing Wu, Xianping Tao, Hung K. Pung, and Jian Lu. epsicar: An emerging patterns based approach to sequential, interleaved and concurrent activity recognition. In *IEEE International Conference on Pervasive Computing and Communications*, pages 1–9, March 2009.

[70] Donghai Guan, Weiwei Yuan, Young K. Lee, Andrey Gavrilov, and Sungyoung Lee. Activity recognition based on semi-supervised learning. In *13th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA 2007)*, pages 469–475, August 2007.

[71] Isabelle Guyon, Bernhard E. Boser, and Vladimir Vapnik. Automatic capacity tuning of very large vc-dimension classifiers. In *Advances in Neural Information Processing Systems 5, [NIPS Conference]*, pages 147–155, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.

[72] Isabelle Guyon and André Elisseeff. An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research (JMLR)*, 3(3):1157–1182, 2003.

[73] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1):389–422, January 2002.

[74] Mohammed M. Hassan, Md. Zia Uddin, Amr Mohamed, and Ahmad Almogren. A robust human activity recognition system using smartphone sensors and deep learning. *Future Generation Computer Systems*, 81:307–313, 2018.

[75] Jian He, Shuang Bai, and Xiaoyi Wang. An unobtrusive fall detection and alerting system based on kalman filter and bayes network classifier. *Sensors*, 17(6), 2017.

[76] Jian He, Mingwo Zhou, Xiaoyi Wang, and Yi Han. A wearable method for autonomous fall detection based on kalman filter and k-nn algorithm. In *IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pages 420–423, October 2016.

[77] Chia-Yeh Hsieh, Chih-Ning Huang, Kai-Chun Liu, Woei-Chyn Chu, and Chia-Tai Chan. A machine learning approach to fall detection algorithm using wearable sensor. In *International Conference on Advanced Materials for Science and Engineering (ICAMSE)*, pages 707–710, Nov 2016.

[78] Abderrazak Iazzi, Mohammed Rziza, and Rachid O. H. Thami. Fall detection based on posture analysis and support vector machine. In *4th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*, pages 1–6, March 2018.

[79] Raul Igual, Carlos Medrano, and Inmaculada Plaza. Challenges, issues and trends in fall detection systems. *BioMedical Engineering OnLine*, 12(1):66–90, July 2013.

[80] Raul Igual, Carlos Medrano, and Inmaculada Plaza. A comparison of public

datasets for acceleration-based fall detection. *Medical Engineering & Physics*, 000:1–9, 2015.

[81] Authoritative information, statistics to promote better health, and wellbeing. Falls in older people. http://www.aihw.gov.au/injury/falls/, 2013. [Online; accessed 22-May-2017].

[82] Timo Jamsa, Maarit Kangas, Irene Vikman, Lars Nyberg, and Raija Korpelainen. Fall detection in the older people: from laboratory to real-life. In *Proceedings of the Estonian Academy of Sciences*, volume 63, pages 253–257, 2014.

[83] Sungmook Jung, Seungki Hong, Jaemin Kim, Sangkyu Lee, Taeghwan Hyeon, Minbaek Lee, and Dae Hyeong Kim. Wearable Fall Detector using Integrated Sensors and Energy Devices. *Scientific Reports*, 5:1–9, 2015.

[84] Satya S. Kambhampati, Vishal Singh, M. S. Manikandan, and Barathram Ramkumar. Unified framework for triaxial accelerometer-based fall event detection and classification using cumulants and hierarchical decision tree classifier. *Healthcare Technology Letters*, 2(4):101–107, 2015.

[85] M. Kangas, I. Vikman, L. Nyberg, R. Korpelainen, J. Lindblom, and T. Jämsä. Comparison of real-life accidental falls in older people with experimental falls in middle-aged test subjects. *Gait & Posture*, 35(3):500–505, 2012.

[86] Maarit Kangas, Antti Konttila, Per Lindgren, Ilkka Winblad, and Timo Jämsä. Comparison of low-complexity fall detection algorithms for body attached accelerometers. *Gait & Posture*, 28(2):285–291, 2008.

[87] Maarit Kangas, Antti Konttila, Ilkka Winblad, and Timo Jämsä. Determination of simple thresholds for accelerometry-based parameters for fall detection. In *29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 1367–1370, Lyon, 2007.

[88] Maarit Kangas, Raija Korpelainen, Irene Vikman, Lars Nyberg, and Timo Jämsä. Sensitivity and false alarm rate of a fall sensor in long-term fall detection in the elderly. *Gerontology*, 61(1):61–68, 2015.

[89] Maarit Kangas, Irene Vikman, Jimmie Wiklander, Per Lindgren, Lars Nyberg, and Timo Jämsä. Sensitivity and specificity of fall detection in people aged 40 years and over. *Gait & Posture*, 29(4):571–574, 2009.

[90] Dean M. Karantonis, Michael R. Narayanan, Merryn Mathie, Nigel H. Lovell, and Branko G. Celler. Implementation of a real-time human movement classifier using a triaxial accelerometer for ambulatory monitoring. *IEEE Transactions on Information Technology in Biomedicine*, 10(1):156–167, 2006.

[91] Lih J. Kau and Chih S. Chen. A smart phone-based pocket fall accident detection system. *IEEE International Symposium on Bioelectronics and Bioinformatics*, 19(1):44–56, 2014.

[92] Lih-Jen Kau and Chih-Sheng Chen. A smart phone-based pocket fall accident detection, positioning, and rescue system. *IEEE Journal of Biomedical and Health Informatics*, 19(1):44–56, January 2015.

[93] V. Kecman. *Support Vector Machines – An Introduction*, pages 1–47. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[94] Adil M. Khan, Young-Koo Lee, Sungyoung Y. Lee, and Tae-Song Kim. A triaxial accelerometer-based physical-activity recognition via augmented-signal features and a hierarchical recognizer. *IEEE Transactions on Information Technology in Biomedicine*, 14(5):1166–1172, September 2010.

[95] Shehroz S. Khan and Babak Taati. Detecting unseen falls from wearable devices using channel-wise ensemble of autoencoders. *Expert Systems with Applications*, 87:280–290, 2017.

[96] J. Klenk, C. Becker, F. Lieken, S. Nicolai, W. Maetzler, W. Alt, W. Zijlstra, J.M. Hausdorff, R.C. van Lummel, L. Chiari, and U. Lindemann. Comparison of acceleration signals of simulated and real-world backward falls. *Medical Engineering & Physics*, 33(3):368–373, 2011.

[97] Joachim Krauth. The interpretation of significance tests for independent and dependent samples. *Journal of Neuroscience Methods*, 9(4):269–281, 1983.

[98] Oscar D. Lara and Miguel a. Labrador. A Survey on Human Activity Recognition using Wearable Sensors. *IEEE Communications Surveys & Tutorials*, 15(3):1192–1209, 2013.

[99] Huong T. Le and Luan V. Tran. Automatic feature selection for named entity recognition using genetic algorithm. In *Proceedings of the Fourth Symposium on Information and Communication Technology*, SoICT '13, pages 81–87, New York, NY, USA, 2013. ACM.

[100] Jung K. Lee, Stephen N. Robinovitch, and Edward J. Park. Inertial sensing-based pre-impact detection of falls involving near-fall scenarios. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 23(2):258–266, March 2015.

[101] Young-Sook Lee and Wan-Young Chung. Visual sensor based abnormal event detection with moving shadow removal in home healthcare applications. *Sensors*, 12(1):573–584, 2012.

[102] Jie Li, Min Li, Zhongya Wang, and Qingying Zhao. An improved classification method for fall detection based on bayesian framework. In *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 237–242, December 2015.

[103] Yun Li, K. C. Ho, and Mihail Popescu. Efficient source separation algorithms

for acoustic fall detection using a microsoft kinect. *IEEE Transactions on Biomedical Engineering*, 61(3):745–755, March 2014.

[104] Shing H. Liu and Wen C. Cheng. Fall detection with the support vector machine during scripted and continuous unscripted activities. *Sensors (Switzerland)*, 12(9):12301–12316, 2012.

[105] Konrad Lorincz, Bor-rong Chen, Geoffrey Werner Challen, Atanu Roy Chowdhury, Shyamal Patel, Paolo Bonato, and Matt Welsh. Mercury: A wearable sensor network platform for high-fidelity motion analysis. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, SenSys '09, pages 183–196, New York, NY, USA, 2009. ACM.

[106] Xin Ma, Haibo Wang, Bingxia Xue, Mingang Zhou, Bing Ji, and Yibin Li. Depth-based human fall detection via shape features and improved extreme learning machine. *IEEE Journal of Biomedical and Health Informatics*, 18(6):1915–1922, November 2014.

[107] Andrea Mannini and Angelo Maria Sabatini. Machine learning methods for classifying human physical activity from on-body accelerometers. *Sensors*, 10(2):1154–1175, 2010.

[108] M. J. Mathie, N. H. Lovell, A. C. F. Coster, and B. G. Celler. Determining activity using a triaxial accelerometer. In *Proceedings of the Second Joint 24th Annual Conference and the Annual Fall Meeting of the Biomedical Engineering Society, Engineering in Medicine and Biology*, volume 3, pages 2481–2482 vol. 3, October 2002.

[109] Carlos Medrano, Raul Igual, Inmaculada Plaza, and Manuel Castro. Detecting falls as novelties in acceleration patterns acquired with smartphones. *PLoS ONE*, 9(4), 2014.

[110] Ludovic Minvielle, Mounir Atiq, Renan Serra, Mathilde Mougeot, and Nicolas Vayatis. Fall detection using smart floor sensor and supervised learning. In *39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 3445–3448, July 2017.

[111] Kyle C. Moylan and Ellen F. Binder. Falls in older adults: risk assessment, management and prevention. *The American Journal of Medicine*, 120(6):493.e1–e6, 2007.

[112] Viet D. Nguyen, Minh T. Le, Anh D. Do, Hoang H. Duong, Toan D. Thai, and Duc H. Tran. An efficient camera-based surveillance for fall detection of elderly people. In *9th IEEE Conference on Industrial Electronics and Applications*, pages 994–997, June 2014.

[113] Qin Ni, Timothy Patterson, Ian Cleland, and Chris Nugent. Dynamic detection of window starting positions and its implementation within an activity recognition framework. *Journal of Biomedical Informatics*, 62:171–180, 2016.

[114] Mohd H. M. Noor, Zoran Salcic, and Kevin I-Kai Wang. Adaptive sliding window segmentation for physical activity recognition using a single tri-axial accelerometer. *Pervasive and Mobile Computing*, 38:41–59, 2017.

[115] N. Noury, A. Fleury, P. Rumeau, A. K. Bourke, G. O. Laighin, V. Rialle, and J. E. Lundy. Fall detection - Principles and Methods. *29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 1663–1666, 2007.

[116] Thomas Nowotny. Two challenges of correct validation in pattern recognition. *Frontiers in Robotics and AI*, 1:5, 2014.

[117] Il-Seok Oh, Jin-Seon Lee, and Byung-Ro Moon. Hybrid genetic algorithms for feature selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1424–1437, 2004.

[118] Olukunle Ojetola. *Detection of Human Falls using Wearable Sensors*. PhD thesis, Coventry University, 2013.

[119] Olukunle Ojetola, Elena Gaura, and James Brusey. Data set for fall events and daily activities from inertial sensors. In *Proceedings of the 6th ACM Multimedia Systems Conference*, MMSys '15, pages 243–248, New York, NY, USA, 2015. ACM.

[120] Olukunle Ojetola, Elena I. Gaura, and James Brusey. Fall detection with wearable sensors - Safe (SmArt Fall dEtection). In *Proceedings 7th International Conference on Intelligent Environments*, pages 318–321, 2011.

[121] World Health Organization. Falls. `http://www.who.int/mediacentre/factsheets/fs344/en/`, 2012. [Online; accessed 1 August 2015].

[122] Javier Ortiz Laguna, Angel García Olaya, and Daniel Borrajo. A dynamic sliding window approach for activity recognition. In Joseph A. Konstan, Ricardo Conejo, José L. Marzo, and Nuria Oliver, editors, *User Modeling, Adaption and Personalization*, pages 219–230, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[123] Natthapon Pannurat, Surapa Thiemjarus, and Ekawit Nantajeewarawat. Automatic fall monitoring: A review. *Sensors*, 14(7):12900–12936, 2014.

[124] Roberto Paoli, Francisco J. Fernández-Luque, Ginés Doménech, Félix Martínez, Juan Zapata, and Ramón Ruiz. A system for ubiquitous fall monitoring at home via a wireless sensor network and a wearable mote. *Expert Systems with Applications*, 39(5):5566–5575, 2012.

[125] Fabian Pedregosa, Gaël Varoquaux, Alexander Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Courna-

peau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[126] Cuong Pham and Tu Minh Phuong. Real-time fall detection and activity recognition using low-cost wearable sensors. In Beniamino Murgante, Sanjay Misra, Maurizio Carlini, Carmelo M. Torre, Hong-Quang Nguyen, David Taniar, Bernady O. Apduhan, and Osvaldo Gervasi, editors, *Computational Science and Its Applications – ICCSA 2013*, pages 673–682, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[127] I P. E. S. Putra, James Brusey, and Elena Gaura. A cascade-classifier approach for fall detection. In *Proceedings of the 5th EAI International Conference on Wireless Mobile Communication and Healthcare*, MOBIHEALTH'15, pages 94–99, ICST, Brussels, Belgium, Belgium, 2015. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[128] Manola Ricciuti, Susanna Spinsante, and Ennio Gambi. Accurate fall detection in a top view privacy preserving configuration. *Sensors*, 18(6), 2018.

[129] Henry Rimminen, Juha Lindström, Matti Linnavuo, and Raimo Sepponen. Detection of falls among the elderly by a floor sensor using the electric near field. *IEEE Transactions on Information Technology in Biomedicine*, 14(6):1475–1476, 2010.

[130] Ramyar Saeedi, Brian Schimert, and Hassan Ghasemzadeh. Cost-sensitive feature selection for on-body sensor localization. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, UbiComp '14 Adjunct, pages 833–842, New York, NY, USA, 2014. ACM.

[131] Sofia Savvaki, Grigorios Tsagkatakis, Athanasia Panousopoulou, and Panagi-

otis Tsakalides. Matrix and tensor completion on a human activity recognition framework. *IEEE Journal of Biomedical and Health Informatics*, 21(6):1554–1561, November 2017.

[132] Kamal Sehairi, Fatima Chouireb, and Jean Meunier. Elderly fall detection system based on multiple shape features and motion analysis. In *International Conference on Intelligent Systems and Computer Vision (ISCV)*, pages 1–8, April 2018.

[133] Shaoming Shan and Tao Yuan. A wearable pre-impact fall detector using feature selection and support vector machine. In *IEEE 10th International Conference On Signal Processing Proceedings*, pages 1686–1689, October 2010.

[134] A. Shrestha, J. Le Kernec, F. Fioranelli, E. Cippitellii, E. Gambi, and S. Spinsante. Feature diversity for fall detection and human indoor activities classification using radar systems. In *International Conference on Radar Systems (Radar 2017)*, pages 1–6, Oct 2017.

[135] S.N. Sivanandam and S. N. Deepa. *Genetic Algorithms*, pages 15–37. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[136] Aino Sorvala, Esko Alasaarela, Hannu Sorvoja, and Risto Myllylä. A two-threshold fall detection algorithm for reducing false alarms. In *6th International Symposium on Medical Information and Communication Technology (ISMICT)*, pages 1–4, March 2012.

[137] Daby Sow, Deepak S. Turaga, and Michael Schmidt. *Mining of Sensor Data in Healthcare: A Survey*, pages 459–504. Springer US, Boston, MA, 2013.

[138] Vianda S. Stel, Johannes H. Smit, Saskia M. F. Pluijm, and Paul Lips. Consequences of falling in older men and women and risk factors for health service use and functional decline. *Age and Ageing*, 33(1):58–65, 2004.

[139] Erik E. Stone and Marjorie Skubic. Fall detection in homes of older adults using the microsoft kinect. *IEEE Journal of Biomedical and Health Informatics*, 19(1):290–301, January 2015.

[140] Angela Sucerquia, Jose David Lopez, and Jesus F. Vargas-Bonilla. SisFall: A Fall and Movement Dataset. *Sensors*, 17(1), 2017.

[141] Angela Sucerquia, José David López, and Jesús Francisco Vargas-Bonilla. Real-life/real-time elderly fall detection with a triaxial accelerometer. *Sensors*, 18(4), 2018.

[142] Timo Sztyler, Heiner Stuckenschmidt, and Wolfgang Petrich. Position-aware activity recognition with wearable devices. *Pervasive and Mobile Computing*, 38:281–295, 2017. Special Issue IEEE International Conference on Pervasive Computing and Communications (PerCom) 2016.

[143] Xinhua Tang, Gianluca Falco, Emanuela Falletti, and Letizia Lo Presti. Complexity reduction of the kalman filter-based tracking loops in gnss receivers. *GPS Solutions*, 21(2):685–699, April 2017.

[144] Shuai Tao, Mineichi Kudo, and Hidetoshi Nonaka. Privacy-preserved behavior analysis and fall detection by an infrared ceiling sensor network. *Sensors*, 12(12):16920–16936, 2012.

[145] Mary E. Tinetti. Predictors and Prognosis of Inability to Get Up After Falls Among Elderly Persons. *JAMA: The Journal of the American Medical Association*, 269(1):65–70, 1993.

[146] Eeva Tuunainen, Jyrki Rasku, Pirkko Jäntti, and Ilmari Pyykkö. Risk factors of falls in community dwelling active elderly. *Auris Nasus Larynx*, 41(1):10–16, 2014.

[147] Huan-Wen Tzeng, Mei-Yung Chen, and Jai-Yu Chen. Design of fall detection system with floor pressure and infrared image. In *2010 International Conference on System Science and Engineering*, pages 131–135, July 2010.

[148] Marcela Vallejo, Claudia V. Isaza, and José D. López. Artificial neural networks as an alternative to traditional fall detection methods. In *35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 1648–1651, Osaka, July 2013.

[149] George Vavoulas, Matthew Pediaditis, Emmanouil G. Spanakis, and Manolis Tsiknakis. The MobiFall dataset: An initial evaluation of fall detection algorithms using smartphones. *13th IEEE International Conference on BioInformatics and BioEngineering*, pages 1–4, 2013.

[150] Sangameswar Venkatraman and Gary G. Yen. A Generic Framework for Constrained Optimization Using Genetic Algorithms. *IEEE Transactions on Evolutionary Computation*, 9(4):424–435, 2005.

[151] Jie Wan, Michael J. O'grady, and Gregory M. O'hare. Dynamic sensor event segmentation for real-time activity recognition in a smart home context. *Personal Ubiquitous Comput.*, 19(2):287–301, February 2015.

[152] Changhong Wang, Wei Lu, Michael R. Narayanan, David C. W. Chang, Stephen R. Lord, Stephen J. Redmond, and Nigel H. Lovell. Low-power fall detector using triaxial accelerometry and barometric pressure sensing. *IEEE Transactions on Industrial Informatics*, 12(6):2302–2311, December 2016.

[153] Changhong Wang, Michael R. Narayanan, Stephen R. Lord, Stephen J. Redmond, and Nigel H. Lovell. A low-power fall detection algorithm based on triaxial acceleration and barometric pressure. In *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 570–573, August 2014.

[154] Changhong Wang, Stephen J. Redmond, Wei Lu, Michael C. Stevens, Stephen R. Lord, and Nigel H. Lovell. Selecting power-efficient signal features for a low-power fall detector. *IEEE Transactions on Biomedical Engineering*, 64(11):2729–2736, November 2017.

[155] Gaojing Wang, Qingquan Li, Lei Wang, Wei Wang, Mengqi Wu, and Tao Liu. Impact of sliding window length in indoor human motion modes and pose pattern recognition based on smartphone sensors. *Sensors*, 18(6), 2018.

[156] Ning Wang, Geoff V. Merrett, Robert G. Maunder, and Alex Rogers. Energy and accuracy trade-offs in accelerometry-based activity recognition. In *22nd International Conference on Computer Communication and Networks (ICCCN)*, pages 1–6, July 2013.

[157] Shuangquan Wang, Jie Yang, Ningjiang Chen, Xin Chen, and Qinfeng Zhang. Human activity recognition with user-free accelerometers in the sensor networks. In *International Conference on Neural Networks and Brain*, volume 2, pages 1212–1217, Beijing, October 2005.

[158] Zhelong Wang, Donghui Wu, Jianming Chen, Ahmed Ghoneim, and Mohammad A. Hossain. A triaxial accelerometer-based human activity recognition via eemd-based features and game-theory-based feature selection. *IEEE Sensors Journal*, 16(9):3198–3207, May 2016.

[159] Zhelong Wang, Donghui Wu, Raffaele Gravina, Giancarlo Fortino, Yongmei Jiang, and Kai Tang. Kernel fusion based extreme learning machine for cross-location activity recognition. *Information Fusion*, 37:1–9, 2017.

[160] Johan Wannenburg and Reza Malekian. Physical activity recognition from smartphone accelerometer data for user context awareness sensing. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(12):3142–3149, December 2017.

[161] Jiahui Wen and Zhiying Wang. Sensor-based adaptive activity recognition with dynamically available sensors. *Neurocomputing*, 218:307–317, 2016.

[162] Huile Xu, Jinyi Liu, Haibo Hu, and Yi Zhang. Wearable sensor-based human activity recognition method with multi-features extracted from hilbert-huang transform. *Sensors*, 16(12), 2016.

[163] Che-Chang Yang and Yeh-Liang Hsu. A review of accelerometry-based wearable motion detectors for physical activity monitoring. *Sensors*, 10(8):7772–7788, 2010.

[164] Jhun-Ying Yang, Jeen-Shing Wang, and Yen-Ping Chen. Using acceleration measurements for activity recognition: An effective learning algorithm for constructing neural classifiers. *Pattern Recognition Letters*, 29(16):2213–2220, 2008.

[165] Jihoon Yang and Vasant Honavar. *Feature Subset Selection Using a Genetic Algorithm*, pages 117–136. Springer US, Boston, MA, 1998.

[166] Lei Yang, Yanyun Ren, Huosheng Hu, and Bo Tian. New fast fall detection method based on spatio-temporal context tracking of head by using depth images. *Sensors*, 15(9):23004–23019, 2015.

[167] Lei Yu and Huan Liu. Efficient Feature Selection via Analysis of Relevance and Redundancy. *Journal of Machine Learning Research*, 5:1205–1224, 2004.

[168] Jian Yuan, Kok K. Tan, Tong H. Lee, and Gerald C. H. Koh. Power-efficient interrupt-driven algorithms for fall detection and classification of activities of daily living. *IEEE Sensors Journal*, 15(3):1377–1387, March 2015.

[169] Aras Yurtman and Billur Barshan. Activity recognition invariant to sensor orientation with wearable motion sensors. *Sensors*, 17(8), 2017.

Appendix A

FARSEEING dataset subjects profiles

Table A.1 shows detailed information of the FARSEEING dataset. The information includes: hardware name, sampling rate, age, gender, height, weight, fall direction, activity before falling (pre-activity), and fall description. Regarding the hardware specification, this dataset does not provide a complete information. Also, some data do not provide the direction of fall.

Table A.1: FARSEEING dataset information

Subject	Device	Sensor placement	Sampling rate (Hz)	Age	Gender	Height	Weight	Fall direction	Pre-activity	Fall description
Subject 1	ActivPAL3	thigh	20	73	Male	171	57	Unknown	Walking	Walking down the corridor, hooked into a handrail then fell down.
Subject 2	MiniMod	L5	100	65	Female	160	74	Backward	Standing	After walking with a wheeled walker, stood behind a chair, then fell backwards on the floor.
Subject 3	ActivPal3	thigh	20	70	Female	148	51	Unknown	Bending	Wanted to bend forward to the ground for picking up an object.
Subject 4	ActivPal3	thigh	20	86	Female	163	53	Unknown	Standing	Wanted to go to toilet, fell down in the bathroom.
Subject 5	ActivPal3	thigh	20	62	Male	190	101	Unknown	Transfer	Wanted to stand up unassisted in the dining room and fell down.
Subject 6	MiniMod	L5	100	71	Female	154	56	Forward	Bending down	Fell forward while bending down to fix the shoelace.
Subject 7	Hybrid	L5	100	72	Male	178	92	Forward	Bending down	Wanted to pick up an object from the ground
Subject 8	Hybrid	L5	100	72	Male	178	92	Forward	Transfer	Got up from the chair and wanted to walk
Subject 9	Hybrid	L5	100	72	Male	178	92	Side forward	Walking	When trying to move to the side, wheeled walker fell forward. Freezing of the movement.
Subject 10	ActivPal3	thigh	20	79	Male	181	86	Sideward	Walking	Walking to the toilet, slipped and fell on the side to the floor.
Subject 11	ActivPal3	thigh	20	79	Male	180	79	Unknown	Transfer	Wanted to pick up the urine bottle, fell down during sit-to-stand transfer.
Subject 12	ActivPal3	thigh	20	78	Male	180	88	Unknown	Transfer	Standing up out of the wheelchair without using brakes, then fell down.
Subject 13	MiniMod	L5	100	65	Female	161	67	Backward	Standing	Went to the table in the dining room. Fell down backwards on the buttock.
Subject 14	MiniMod	L5	100	65	Female	161	67	Backward	Standing	Person held on the wall, then fell down backwards on the buttock.
Subject 15	MiniMod	L5	100	60	Male	173	74	Unknown	Walking	Fell during walking.
Subject 16	MiniMod	L5	100	60	Male	173	74	Unknown	Walking	Walking, then fell down in front of the entrance of the house.
Subject 17	Hybrid	L5	100	75	Female	159	75	Backward	Standing	Changing the hip protector, fell backwards on the ground and hit the toilet.
Subject 18	MiniMod	L5	100	56	Female	150	81	Unknown	Walking	Fell down in front of the entrance of the house.
Subject 19	MiniMod	L5	100	56	Female	150	81	Unknown	Walking	Walking, then fell down opening the door in the entrance hall.
Subject 20	Hybrid	L5	100	71	Female	163	75	Backward on the left	Walking	Walking to the bathroom, stopped with freezing and fell from standing position.
Subject 21	Hybrid	L5	100	71	Female	163	75	Backward	Standing	Standing at the wardrobe, wanted to walk backwards and then fell on buttocks.
Subject 22	Hybrid	L5	100	71	Female	163	75	Backward	Standing	Walking backwards from the wash basin, then fell backwards.

207

Appendix B

Publications, presentations, and attended conferences

The following outputs have been presented/published/accepted/submitted to symposium/conferences/journal/competition.

B.0.1 Conference proceedings and poster

A cascade-classifier approach for fall detection.

I Putu Edy Suardiyana Putra. Presented as a research poster in the Coventry University Research Symposium, 2014.

A Cascade- Classifier Approach for Fall Detection

I Putu Edy Suardiyana Putra, James Brusey, and Elena Gaura. In Proceedings of the 5th EAI International Conference on Wireless Mobile Communication and Healthcare (MOBIHEALTH'15), Akram Alomainy, William Whittow, Yang Hao, Konstantina S. Nikita, and Clive G. Parini (Eds.). ICST (Institute for Computer Sciences, Social-Informatics and Telecom- munications Engineering), ICST, Brussels, Belgium, Belgium, 94-99.

An intelligent system for fall detection using wearable sensors: issues and challenges

I Putu Edy Suardiyana Putra, James Brusey, and Elena Gaura. In Proceedings of the 10th International Student Conference on Advanced Science and Technology (ICAST) 2015, Surabaya, Indonesia, pp 93-94.

Genetic algorithm-based feature selection technique for fall detection using multi-placement wearable sensors

I Putu Edy Suardiyana Putra and Rein Vesilo. In Proceedings of the 12th International conference on Body Area Networks 2017.

Window-size impact on detection rate of wearable sensor-based fall detection using supervised machine learning

I Putu Edy Suardiyana Putra and Rein Vesilo. In proceedings of the 1st IEEE Life Sciences Conference 2017.

B.0.2 Journal publication

An event-triggered machine learning approach for fall detection using wearable sensors

I Putu Edy Suardiyana Putra, James Brusey, Elena Gaura, and Rein Vesilo. Published in MDPI Sensors Journal.

B.0.3 Presentations

- **The inaugural Macquarie University research minds showcase** (2016) – Macquarie University, Sydney, Australia.

- **Sydney research bazar** (ResBaz 2017) – University of Technology Sydney, Sydney, Australia.

B.0.4 Awards

- **Macquarie University Postgraduate Research Funding (**PGRF 2017**)** – Macquarie University, Sydney, Australia.

Appendix C

Ethical approval

The work in this thesis has been approved through Coventry Universities ethical approval process. The ethical approvals follows:

Low Risk Research Ethics Approval

Project Title

Fall Detection Algorithm Development

Record of Approval

Principal Investigator

I **request an ethics peer review** and confirm that I have answered all relevant questions in this checklist honestly.	X
I confirm that I will carry out the project in the ways described in this checklist. I will immediately suspend research and request new ethical approval if the project subsequently changes the information I have given in this checklist.	X
I confirm that I, and all members of my research team (if any), have read and agreed to abide by the Code of Research Ethics issued by the relevant national learned society.	X
I confirm that I, and all members of my research team (if any), have read and agreed to abide by the University's Research Ethics, Governance and Integrity Framework.	X

Name: I Putra .....

Date: 06/11/2014.....

Student's Supervisor (if applicable)

I have read this checklist and confirm that it covers all the ethical issues raised by this project fully and frankly. I also confirm that these issues have been discussed with the student and will continue to be reviewed in the course of supervision.

Name: James Brusey .....

Date: 27/11/2014.....

Reviewer (if applicable)

Date of approval by anonymous reviewer: 30/07/2015

Low Risk Research Ethics Approval Checklist

Project Information

Project Ref	P28811
Full name	I Putra
Faculty	Faculty of Engineering and Computing
Department	Aviation, Aerospace and Electrical
Supervisor	James Brusey
Module Code	ECAAE
EFAAF Number	
Project title	Fall Detection Algorithm Development
Date(s)	27/05/2015 - 31/12/2015
Created	06/11/2014 12:22

Project Summary

This research will develop an automated fall detection system for the elderly. A number of such systems have been proposed, with claims of fall detection accuracy of over 90% based on accelerometers and gyroscopes. However, most such fall detection algorithms have been developed based on observational analysis of the data gathered, leading to thresholds setting for fall/non-fall situations. Whilst the fall detection accuracies reported in prior studies appear to be high, there is a little evidence that the threshold based methods proposed generalise well with different subjects and different data gathering strategies or experimental scenarios. Moreover, some studies show that postural sway become an pre-impact indicator. However, there are few studies that investigate the use of wearable sensors to detect postural sway. Thus, this research will develop machine learning algorithms to discriminate between falls, postural sway, and Activities of Daily Living (ADL). Secondary dataset will be used in the development process.

Names of Co-Investigators and their organisational affiliation (place of study/employer)	
Is the project self-funded?	YES
Who is funding the project?	Coventry University
Has the funding been confirmed?	YES
Are you required to use a Professional Code of Ethical Practice appropriate to your discipline?	NO
Have you read the Code?	NO

Project Details

What is the purpose of the project?	The purpose of this project is to design a system that uses machine learning algorithms to discriminate between falls and ADL. Using acceleration and gyroscopic sensors, the movement, rotation and orientation of the body can be measured. A secondary dataset will be used for training and testing the algorithm.
What are the planned or desired outcomes?	A machine learning based fall detection algorithms will be developed and evaluated for both offline and real-time applications.
Explain your research design	The research will be started by analysing the raw data from the dataset using some softwares. Then, the information from that analysis will be used for developing the machine learning algorithm. The developed algorithm will be tested through some simulations.
Outline the principal methods you will use	The method that will be used in this research is: 1: Review literatures to keep abreast of state of the art. 2: Analysis the raw data from dataset. 3: Develop a machine learning algorithm for fall detection. 4: Run some simulations for testing. 5: Analyse the result and write a report about it.

Are you proposing to use an external research instrument, validated scale or follow a published research method?	NO
If yes, please give details of what you are using	
Will your research involve consulting individuals who support, or literature, websites or similar material which advocates, any of the following: terrorism, armed struggles, or political, religious or other forms of activism considered illegal under UK law?	NO
Are you dealing with Secondary Data? (e.g. sourcing info from websites, historical documents)	YES
Are you dealing with Primary Data involving people? (e.g. interviews, questionnaires, observations)	NO
Are you dealing with personal or sensitive data?	YES

Is the project solely desk based? (e.g. involving no laboratory, workshop or off-campus work or other activities which pose significant risks to researchers or participants)	YES
Are there any other ethical issues or risks of harm raised by the study that have not been covered by previous questions?	NO
If yes, please give further details	

External Ethical Review

Question		Yes	No
1	Will this study be submitted for ethical review to an external organisation? (e.g. Another University, Social Care, National Health Service, Ministry of Defence, Police Service and Probation Office)		X
	If YES, name of external organisation		
2	Will this study be reviewed using the IRAS system?		X
3	Has this study previously been reviewed by an external organisation?		X

Risk of harm, potential harm and disclosure of harm

Question			Yes	No
1	Is there any significant risk that the study may lead to physical harm to participants or researchers?			X
	If YES, please explain how you will take steps to reduce or address those risks			
2	Is there any significant risk that the study may lead to psychological or emotional distress to participants?			X
	If YES, please explain how you will take steps to reduce or address those risks			
3	Is there any risk that the study may lead to psychological or emotional distress to researchers?			X
	If YES, please explain how you will take steps to reduce or address those risks			
4	Is there any risk that your study may lead or result in harm to the reputation of participants, researchers, or their employees, or any associated persons or organisations?			X
	If YES, please explain how you will take steps to reduce or address those risks			
5	Is there a risk that the study will lead to participants to disclose evidence of previous criminal offences, or their intention to commit criminal offences?			X
	If YES, please explain how you will take steps to reduce or address those risks			
6	Is there a risk that the study will lead participants to disclose evidence that children or vulnerable adults are being harmed, or at risk or harm?			X
	If YES, please explain how you will take steps to reduce or address those risks			
7	Is there a risk that the study will lead participants to disclose evidence of serious risk of other types of harm?			X
	If YES, please explain how you will take steps to reduce or address those risks			
8	Are you aware of the CU Disclosure protocol?		X	

Online and Internet Research

Question		Yes	No
1	Will any part of your study involve collecting data by means of electronic media (e.g. the Internet, e-mail, Facebook, Twitter, online forums, etc)?		X
	If YES, please explain how you will obtain permission to collect data by this means		
2	Is there a possibility that the study will encourage children under 18 to access inappropriate websites, or correspond with people who pose risk of harm?		X
	If YES, please explain further		
3	Will the study incur any other risks that arise specifically from the use of electronic media?		X
	If YES, please explain further		
4	Will you be using survey collection software (e.g. BoS, Filemaker)?		X
	If YES, please explain which software		
5	Have you taken necessary precautions for secure data management, in accordance with data protection and CU Policy?	X	
	If NO, please explain why not		

Certificate of Ethical Approval

Applicant:

I Putra

Project Title:

Fall Detection Algorithm Development

This is to certify that the above named applicant has completed the Coventry University Ethical Approval process and their project has been confirmed and approved as Low Risk

Date of approval:

30 July 2015

Project Reference Number:

P28811

Human Ethics Application

Application ID :	5201600506
Application Title :	Human activity recognition and fall detection using wearable sensors
Date of Submission :	N/A
Primary Investigator :	Mr I Putu Edy Suardiyana Putra

Before you begin

Human Ethics

Ethics application type*

Human

MQ Application ID

5201600506

I have read and understood each of the guidelines below.*

- ○ No
- ◉ Yes

This online form applies to all human research ethics applications requiring submission to Macquarie University (MQ). As a result, some questions, pages and sections will open or close based on your answers as you complete the form.

Further information is available in the **user guide for human ethics applicants**. For all other forms and templates, please visit the **Forms** page at the MQ Human Ethics website.

Please familiarise yourself with the ***National Statement on Ethical Conduct in Human Research (2007) (Updated March 2014)*** as well as the ***Australian Code for the Responsible Conduct of Research (2007)*** (if you have not already done so).

Please consider allowing pop-up windows for this site before continuing with your application.

If you are unfamiliar with any **acronyms or abbreviations** in this form, simply roll your mouse onto the text to reveal the full version or access the **MQ Glossary**.

If you run into any problems while completing this application, please try using another browser before contacting MQ for help.

For all other technical issues, please contact:
(web) **OneHelp (MQ's IT Help Desk)**
(email) **help@mq.edu.au**
(phone) +61 2 9850 4357 (HELP)

For issues specific to the MQ ethics application process:
(web) **Human Ethics FAQs**
(email) **ethics.secretariat@mq.edu.au**
(phone) +61 2 9850 4459.

Before attempting to submit your application, please complete all mandatory questions (marked with a *) and ensure that you have not made any conflicting checklist responses (e.g. selecting 'None of the above' and 'All of the above' at the same time).

You can submit your application at any time, however applications will not be processed between November 30 and February 1 of each year. If this form has not been submitted within six (6) weeks of the creation date (and you have not submitted an appropriate extension request), the project will be considered withdrawn and you will be required to resubmit the project as a new application.

Once ethics approval has been confirmed (i.e. via an approval letter attached to this application form), the Principal Investigator (or Coordinating Investigator) is required to inform MQ if their approved research project is discontinued before the expected date of completion, and to give reasons for this change. This is in accordance with best practice models of ethical review (see *National Statement* items 5.5.6 to 5.5.8). Please refer to the relevant section in the **user guide for human ethics applicants** for further information on this process.

Click the green arrows (at the top and bottom of the screen) to move throughout the application.

1. Key aspects

Title

Department of application creator
*This field is automatically populated and represents the department associated with the initial applicant (or creator) of this form.**

Engineering

1.1 What is the formal title of this research proposal?*

Human activity recognition and fall detection using wearable sensors

1.1.1 Abbreviated title (if applicable)

This question is not answered.

1.2 Has this proposal been previously submitted to, or previously approved by, a Macquarie University HREC?*

- ○ Yes
- ◉ No

1.3 What is the main purpose of this research?*

- ○ Staff research
- ◉ Student research

1.3.1 Please indicate the level of study being undertaken by the student(s).*

- ☐ Undergraduate
- ☐ Honours
- ☑ Postgraduate

1.3.2p Which of the following study options are relevant to the student(s) involved?*

- ☑ Doctor of Philosophy (PhD)
- ☐ Other doctoral degree(s)
- ☐ Master of Philosophy (MPhil)
- ☐ Master of Research (MRes)
- ☐ Masters by coursework
- ☐ Other postgraduate option(s)

2. Research personnel

Personnel details

To begin adding personnel:

- Place the cursor in the textbox and **type the MQ OneID (or full name) that you wish to search for**
- **Click on the magnifying glass** to begin the search
- (If applicable) select the appropriate person from the list provided or, if they do not appear in the list, please select "Add External Person"
- **Respond to all of the mandatory questions/fields** in that person's record
- **Click on the green tick** to add the record to your personnel list (and repeat the process as necessary).

2.1 Personnel table *(please open, complete and save each record)**

1	Full name	Mr I Putu Edy Suardiyana Putra
	Qualifications and relevant experience	Qualifications: - Bachelor degree in Computer Science Major - Master degree in Computer Science Major - Currently I am pursuing my PhD under Co-tutelle agreement between Coventry University, United Kingdom and Macquarie University, Australia. Experience: - I was a junior lecture at Faculty of Computer Science, Universitas Indonesia from 2013-2014. - I was a junior researcher at Faculty of Computer Science, Universitas Indonesia from 2013-2014. - I was a project leader for a project called m-SHAKERS. This project worked on creating a smart system to detect an earthquake using smartphone. - I have been working on wearable sensors since 2014. I started my work in wearable sensor technology at Cogent Lab, Coventry University, United Kingdom.
	Position	Student
	Primary contact?	Yes
	Type	Student
	Given name (in database)	I Putu Edy Suardiyana
	Surname (in database)	Putra
	Current email address (in database)	i-putu-edy-suardiyana.putra@students.mq.edu.au
	Dept/affiliation	4321
	Faculty	4000

Primary contact (selected in Personnel table)
This field will be completed by the system when a primary contact has been assigned in question 2.1.

Mr I Putu Edy Suardiyana Putra

Before progressing to the next page, please ensure you have opened each research personnel record and completed all of the mandatory fields appropriately.

3. Ethical review

Purpose of review

3.1 Regarding the ethical oversight to be provided by MQ, **at which locations will the research be conducted**? *Please select all that apply.**

- [ ] Not applicable - external review only
- [x] Macquarie University (including MGSM, City Campus, etc.)
- [ ] At participants' place of work
- [ ] Public schools
- [ ] Private/independent schools
- [ ] Other locations in Australia
- [ ] Overseas (i.e. not in Australia)
- [ ] Other universities in Australia (one or more)
- [ ] Macquarie University Hospital
- [ ] Other private hospitals in Australia (one or more)
- [ ] Private health practices/clinics in Australia (one or more)
- [ ] Public hospitals in Australia (one or more)
- [ ] Other public health/justice health sites in Australia (one or more)
- [ ] Australian Department of Defence sites (one or more)
- [ ] In participants' homes

Personnel (other)

3.1.3 Is it intended that other people, not yet known, will play a specified role in the conduct of this research project?*

- ○ Yes
- ◉ No

3.1.4 Do the research personnel (including any students) or others involved in any aspect of this research project require **any additional training** in order to undertake this research?*

- ○ Yes
- ◉ No

External committees

3.2 Has this research proposal been submitted to any external HRECs?*

- ○ Yes
- ◉ No

3.2.4.1a Please provide the research start date for which you are asking a **Macquarie University** HREC to provide ethical review. *(dd/mm/yyyy)**

01/09/2016

3.2.4.1b Please provide the research end date for which you are asking a **Macquarie University** HREC to provide ethical review. *(dd/mm/yyyy)**

01/08/2017

3.3 Do you intend to submit this research proposal (or some variation of it) to any other HRECs?*

- ○ Yes
- ◉ No

4. Funding & support

Funding sources

4.1 Which of the following characterises the type(s) of funding being utilised by this research proposal?*

- [ ] Grant(s) - External competitive
- [ ] Grant(s) - Internal competitive
- [ ] Sponsor(s)
- [x] Department/Faculty funding
- [ ] Employer/organisation funding
- [ ] Still seeking funding
- [ ] No funding

4.2 Will the project be supported in other ways? *e.g. in-kind support or equipment by an external party**

- ○ Yes
- ◉ No

Other interests

4.8 Have conditions been imposed upon the use, publication or ownership of the results including the review of data, manuscript draft or scientific presentation by any party?*

○ Yes
◉ No

Conflicts of interest

The NHMRC defines **conflict of interest** (in a research context) as a situation:

- "where a person's individual interests or responsibilities have the potential to influence the carrying out of his or her institutional role or professional obligations in research; or
- where an institution's interests or responsibilities have the potential to influence the carrying out of its research obligations."

4.9a In undertaking this research at MQ, do any conflict of interest issues arise?*

○ Yes
◉ No

5. Nature of research

Summary

5.1 **Be sure to save your content regularly (using the "Save" icon) to avoid losing any information** - the system will timeout after one (1) hour of inactivity.

Please describe the project using lay terms, including:

- aims of the project
- hypothesis/research question
- research methods, and
- research plan.

Please provide references as evidence that the study has a clear rationale.

*Please refrain from using abbreviations and acronyms without providing the full version in the first instance (e.g. 'National Health and Medical Research Council (NHMRC)' subsequently becomes 'NHMRC').**

Aims : Falls are a serious problem in Australia where around 30% of adults over 65 experiencing at least one fall per year, according to NSW Health 2010. Falls account for 40% of injury-related deaths and 1% of total deaths in this age group. This is set to increase as Australia¿s population ages with the proportion of people aged over 65 predicted to increase from 14% (3 million people) in 2010 to 23% (8.1 million people) in 2050. As the number of elderly is increasing while the number of health care services is limited, the need of a monitoring system, especially fall detection system, is an urgent matter to reduce the loads of healthcare services provider without sacrificing the quality of the services.

Traditionally, medical alert systems in a shape of pendant or button that must be pressed by elderly to get the emergency assistance were used. However, these systems are ineffective under certain circumstances (e.g. fainted, unconscious, paralyzed, etc.) when the person are unable to press the button. The current systems are more advance as they are equipped with technology that can automatically detect falls and send emergency alerts. Nevertheless, these advanced systems usually produce a high number of false alarms. Theses issues result in the need of more reliable automated fall detection systems.

The aim of this project is to design and develop In-Suit (Intelligent suit and assistive technology): a wearable sensor-based system for activity recognition and emergency assistance (i.e. during critical events such as falling).The proposed technology consists of wearable wireless sensors combined with an artificial intelligence algorithm. Considering the accuracy and convenience of the subject, the wearable sensors will be integrated into the daily attires, i.e., shirt, belt, and pants. The sensors will detect the activity of the subjects, send it wirelessly to be recognised by the artificial intelligence algorithm on the server accurately.

Hypothesis/ Research questions:
1. Can the proposed system distinguish between fall and activities of daily living?
2. What is the best placement of the sensors to attain the optimum accuracy for fall detection?
3. What is the minimum number of sensors needed to get the optimum accuracy for fall detection?

Research method: This study is a pilot study for developing a smart fall detection system using wearable sensors. This research uses experimental design. The experiment will involve several human subjects to wear the clothes equipped with the sensors.

Research plan: The experiment is conducted in two parts. The first part will involve several subjects to wear the clothes (shirt, pants, and belt) that have equipped with the sensors. Then, the subject will be asked to do their daily activities as natural as possible. The researcher will monitor the subject from distance to make sure there is no hardware/software failure during the session. The second part will involve several subjects to stage some falls on the soft mattress/ air bed. The subject will be requested to wear the clothes (shirt, pants, and belt) and they are required to perform several artificial falls including: fall forward, fall backward, fall on the right side, and fall on the left side on a soft mattress/ air bed. During this session, the subjects will be fully monitored by the researcher. The participants also will be equipped with head, wrist, elbow, and knee protectors during this session.Some additional information will be provided in the attachment.

5.2 Please describe the participation details using lay terms, including:

- what participation will involve
- the projected number, sex and age range of participants, and
- any inclusion and exclusion criteria.

*

For the purpose of this study, 5 people of young and healthy participants from [faculty of science engineering] Macquarie University will be recruited with the following criteria:
- He/she must be an adult (20-30 years old) with no physical impairment.
- He/she must be able to independently provide informed consent

Type of research

5.3 The nature of this project is most appropriately described as involving: *(please select all that apply)**

- [ ] Questionnaires, surveys, interviews and/or focus groups
- [ ] Film, audio and/or video
- [ ] Exposure to ionising radiation
- [ ] Use of medical imaging/equipment
- [ ] Gametes or use/creation of embryos
- [ ] Other
- [x] Participant observation
- [ ] Databanks
- [ ] A clinical trial
- [ ] Clinical research
- [ ] Collection/use of human biospecimens
- [ ] Genetic testing/research
- [ ] A cellular therapy
- [ ] Workplace practices/relationships

5.4 *This question asks whether the true purpose of the research will be concealed from the participants to any extent.*

Does the research with participants involve:*

- [ ] Limited disclosure
- [ ] Active concealment
- [ ] Planned deception
- [x] None of the above

Obtaining information

5.5 What method(s) will be used to obtain participants' consent?*

- [x] Written consent (signed Participant Information and Consent Form)
- [ ] Consent via return email
- [ ] Online consent (selecting relevant questionnaire response option)
- [ ] Oral consent (e.g. face-to-face, telephone or audio/visual recording)
- [ ] Conduct implying consent (e.g. return of a survey)
- [ ] None, I will not be obtaining consent

5.5.1 Please provide further details about how these methods will be used to inform participants (or those deciding for them) about the nature of the project.*

At first the participant information sheet will be given to the participant to read. Then the researcher will also explain the experiment and it's purpose. The participant who is willing to be involved in the study is required to inform the researcher at i-putu-edy-suardiyana.putra@students.mq.edu.au or at +61 (0) 410 254 489. Then, he/she is requested to sign the Consent Form. The signed consent form can be given to the researcher via email or the participant can give it directly.

5.5.2 Please indicate whether the source of the information about participants which will be used in this research project will involve:*

- [x] Collection directly from the participant
- [ ] Collection from another person about the participant
- [ ] Use or disclosure of information by an agency/authority/organisation other than your own
- [ ] Use of information collected previously by you/your organisation for a different purpose/proposal

Benefits & risks

5.7 ***Benefits*** *of research may include, for example, gains in knowledge, insight and understanding, improved social welfare and individual wellbeing, and gains in skill or expertise for individual researchers, teams or institutions.*

What expected benefits (if any) will this research have for the wider community?*

Mainly the elderly people will be benefited. This system can be implemented in age-care, nursing homes, retirement villages where few staffs are available to take care of many elderly people. This system will alert the staffs automatically if any elderly person needs help when they fall unattended.

5.8 What expected benefits (if any) will this research have for participants?*

The expected benefits for the participants is to be able to try the prototype and hence to gain an insight on the emerging wearable technology and how it works. Furthermore, another advantage for the participants is the opportunity in participating in the development of future technology of the health monitoring system.

5.9 *A **risk** is a potential for harm, discomfort or inconvenience and involves:*

- *the likelihood that a harm (or discomfort or inconvenience) will occur, and*
- *the severity of the harm, including its consequences.*

Are there any risks to participants as a result of participation in this research project?*

- (•) Yes
- ( ) No

5.10 Are there any other risks involved in this research? *e.g. to the research team, the institution/site, others, etc.**

- ○ Yes
- ◉ No

5.10.1 **You answered "Yes" to question 5.7p, 5.9 and/or 5.10.**

What are these risks?*

Participants will be required to perform several falls including forward fall, backward fall, left-side fall, and right-side fall. There is minimal risk of injury associated with these activities.

5.10.2 Please explain how these risks will be minimised and managed.*

To minimise the risk of being injured during the session, the participants will be required to perform falls on the soft mattress/air bed to reduce the impact between the surface and the participants' body. Furthermore, the participants will be required to wear head, wrist, elbow, and knee protectors to avoid physical harms. The risk of physical discomfort will be managed by close monitoring of the participant during the experiment. The participants are encourage to inform the researcher whenever they feel discomfort or pain. The participants are also allowed to withdraw their participation without any consequence.

5.10.3 Please explain how these risks will be monitored.*

During the falls experiment, all participants will be monitored closely by the researcher. If any issues arise, the session will be stopped immediately.

5.10.4 Please explain how any harm to participants - resulting from these risks - will be reported.*

Any harms to the participants will be reported directly to to our collaborators from Macquarie Hospital: Ruth Green (she is nursing co-ordinator for orthopaedics and Bone & Joint programs at Macquarie University Hospital) and A/Prof. Desmond Bokor (Associate Professor in Orthopaedic Surgery).

Monitoring

5.11 Please indicate who will be primarily responsible for dealing with any unexpected events (e.g. serious adverse events, etc.).*

- ◉ Research personnel (from this application)
- ○ Other

5.11.1 Which of the research personnel will be primarily responsible for dealing with any unexpected events?*

- ◉ Mr I Putu Edy Suardiyana Putra

5.11.3 Please provide details of this person's experience in managing risks and adverse events.*

The researcher has an experience about conducting research in activity recognition study using wearable sensors at Cogent Labs, Coventry University, UK. The researcher also will contact A/Prof. Desmond John Bokor and Miss Ruth Green from Macquarie University Hospital when the issues arise. Ruth Green is nursing co-ordinator for orthopaedics and Bone & Joint programs at Macquarie University Hospital and Desmond John Bokor is Associate Professor in Orthopaedic Surgery and the head of Orthopaedics and Musculoskeletal Medicine at The Australian School of Advanced Medicine, Macquarie University. Both are health and medical specialists who have experiences in handling fall-related injuries.

5.12 How will the researchers monitor the progress and conduct of the project?*

The researcher will monitor the participants during the sessions. For the first session, the researcher will monitor the participants from distance to avoid any interferences that are caused by the monitoring activities. For the second session, the researcher will closely monitor the participants. All incidents will be reported by A/Prof. Rein (Chief Investigator) using online form at : http://staff.mq.edu.au/human_resources/health_and_safety/ . Any serious injuries will be reported to HREC using the appropriate form within 72 hours.

6. Participant Information and Consent

Participant sample

6.1 Please indicate which of the following 'types of research participants' are likely to be included in your research proposal due to the project design.*

- [ ] Women who are pregnant and/or the human foetus
- [ ] Children and/or young people (i.e. under 18 years)
- [ ] Refugees or asylum seekers
- [ ] Members of the Police Force
- [ ] Members of the Defence Force
- [ ] People highly-dependent on medical care
- [ ] People with a cognitive impairment, an intellectual disability or a mental illness
- [ ] People who may be involved in illegal activity
- [ ] Aboriginal or Torres Strait Islanders
- [ ] People residing outside Australia
- [ ] People whose primary language is not English
- [ ] None of the above
- [ ] Wards of state
- [x] MQ students, MQ staff, MUH staff and/or MUH patients
- [ ] People in teacher-student relationships
- [ ] People in employer-employee relationships
- [ ] People with chronic conditions/disabilities and their carers
- [ ] Healthcare professionals and their patients/clients
- [ ] People in other professional-client relationships
- [ ] Prisoners or detainees

6.2 What is the expected maximum number of total participants to be involved in this project at all sites under review by MQ? *If you cannot provide an exact number, an approximate number will suffice.**

5.00

6.2.1a What is the age range of participants involved in this study?

Lower age limit (in years)*

20.00

6.2.1b Upper age limit (in years)*

30.00

6.2.2 Are there any other relevant characteristics of the participants involved in this study?*

- ( ) Yes
- (•) No

Recruitment methods

6.3 Who will be involved in the recruitment of participants?*

- [x] Research personnel (from this application)
- [ ] Organisation(s)
- [ ] Other

6.3.1 Which of the research personnel will be involved in participant recruitment?*

- [x] Mr I Putu Edy Suardiyana Putra

6.3.1.1 Does recruitment involve a direct personal approach from the research personnel to the potential participants?*

- (•) Yes
- ( ) No

6.3.1.2 Please list the precautions that will be taken to minimise any real or perceived pressure on individuals to enrol.*

An email will be sent to a faculty mailing list to explain the details of the study to candidates. Students who are willing to voluntarily participate are advised to contact the researcher via email. Only the students who are willing to participate will be involved with the experiment. The participants will be requested to read the participant information sheet before signing the consent form. There is no consequence in rejecting the request for being the participant in this study. There is no consequence in withdrawing the participation if the participant feels discomfort.

Recruitment approach

6.3.4 Does recruitment involve the circulation, publication and/or use of:*

- [ ] Advertisement or flyer
- [x] Email or letter
- [ ] Social media
- [ ] Telephone call
- [ ] Website
- [ ] None of the above

6.3.4.2 Please indicate how often/for how long it will be published/distributed.*

The email will be sent once every 3 days to the faculty mailing list.

6.3.5 Please describe how and where the initial contact will be made with potential participants.*

The recruitment will be done by asking the candidates by person. The researcher will come to the candidates by person and explaining the experiment's procedure and asking their willingness to participate in this study. The participants will also be given the researcher's contact details (e-mail and phone number) if it is needed.

6.3.6 Will participants be involved in any related studies?*

- ( ) Yes
- (•) No

Consent methods

6.4 Will consent be obtained for all participants involved in the research?
*Where relevant, this includes consent from parents of children and/or young people, from legal guardians, and from people responsible for participants**

- (•) Yes
- ( ) No

6.5 If a participant (or person on behalf of a participant) chooses not to participate or decides to withdraw from the research, are there specific consequences they should be aware of prior to making these decisions?*

- ( ) Yes
- (•) No

6.6 Is there a possibility that individual participants may be identifiable by others and thereby be exposed to risks?*

- ( ) Yes
- (•) No

Consent specifics

6.7 Do you intend to share the data/tissue collected as part of this research project for any future HREC-approved research?
*i.e. future research projects that may be conducted by other researchers and/or personnel from this project**

- (•) Yes
- ( ) No

6.7.1 What type of consent will you be obtaining?*

Extended (use of data/biospecimens in extensions of this original project/similar future research)

6.7.1.1 Do you wish to deposit the original data/biospecimens collected as part of this research project into a databank/biobank?*

- ( ) Yes
- (•) No

6.8 Will participants receive any financial or other benefits as a result of participation?*

- ( ) Yes
- (•) No

7. Participant specifics

Participant types

7.5.1 **You indicated that your research will involve participants who are in dependent or unequal relationships (as selected on the "Participant sample" page).** For more details, refer to Chapter 4.3 of the *National Statement on Ethical Conduct in Human Research* (**People in dependent or unequal relationships**).

Regarding these participants, please describe the steps that will be taken to ensure that each participant's consent and participation in the project is free and voluntary.*

The participant will be informed that their participation in this project is voluntary and free before they sign the consent form.

7.5.2 Will there be any specific risks to these participants as a result of their dependent/unequal relationship(s)?*

- ○ Yes
- ◉ No

8. Privacy & confidentiality

Privacy and access

8.1.1 Please describe the information that will be collected directly from participants.*

The sensor readings will be collected during the session. Some personal information (such as: age, weight, and height) will be collected as well. A camera will be used to record the fall activities.

8.1.1.1 The information collected by the research team about participants will be in the following form(s):*

- ☑ Individually-identifiable
- ☐ Re-identifiable
- ☐ Non-identifiable

8.1.1.2 Please explain why this information is being collected in individually-identifiable or re-identifiable form.*

The personal information from the participant will help the researcher to validate the experiment result.

8.1.5 As this project will be making information available for future HREC-approved research (as indicated in "**Consent specifics**"), please indicate the form(s) in which this participant information will be shared.*

- ☐ Individually identifiable
- ☐ Re-identifiable
- ☑ Non-identifiable

8.2 Please indicate which of the research personnel who - for the purposes of this research - will have access to the information or authority to use the information.

- ☑ Mr I Putu Edy Suardiyana Putra

8.2.1 Are there any others *(e.g. student supervisors, research monitors, pharmaceutical company monitors)* who - for the purposes of this research - will have authority to use or have access to the information.*

- ◉ Yes
- ○ No

8.2.1.1 Please name these other people.*

A/Prof. Rein Vesilo, A/Prof. Desmond John Bokor, and Ruth Green

8.2.2 Please describe the nature of the access or use for each person.*

A/Prof. Rein Vesilo is the supervisor for this study. A/Prof. Desmond John Bokor and Ruth Green are the advisors of this project.

Storage & disposal

8.3.1 In what formats will the information be stored during the research project? *(e.g. paper copy, computer file on USB, audio device, etc.)*

The data from sensor readings will be stored in digital format and will be stored on computer file. The consent form and participant profile's information (age, height, weight, and fall history) will be stored on paper copy.

8.3.2 Please provide details about where and how the hard and/or electronic copies of data will be securely stored during the project (i.e. to ensure the security of information from misuse, loss, or unauthorised access).*

The data will be stored on the researcher's computer with password protected. The paper copy-based information will be stored in a researcher's drawer and the only key is held by the researcher.

8.3.3 Please provide details about where and how the hard and/or electronic copies of data will be securely stored after completion of the project (i.e. to ensure the security of information from misuse, loss, or unauthorised access).*

The paper copy-based information will be converted to digital form. Then, the digitised participant personal information will be stored together with the data from sensor readings. Both information will be stored on researcher's personal computer with password protected.

8.4.1 As per the *Australian Code for the Responsible Conduct of Research*, are you planning to retain the research data for the **minimum period of five (5) years from the date of publication** (or longer)?*

- ◉ Yes
- ○ No

8.4.1.1 For how long will the information be stored after the completion of the project and why has this period been chosen?*

5 years. The data is very valuable for further research in fall detection.

8.4.2 Are there plans to formally dispose of the research data (in all forms)?*

- ○ Yes
- ◉ No

Reporting results

8.5.1 Is it intended that the results of individual participants from the research will be reported to those participants?*

- ○ Yes
- ◉ No

8.5.1.2a Please indicate which of the research personnel will be responsible for communicating the project results to participants.*

- ☑ Mr I Putu Edy Suardiyana Putra

8.5.1.3 Please explain why the results will not be reported to participants.*

The aims of this study is to develop the accurate and reliable system to monitoring elderly. Therefore, reporting the results to the participants, who are young and healthy, is inappropriate.

8.5.2 Is the research likely to produce information of personal significance to individual participants?*

- ○ Yes
- ◉ No

8.5.3 As this research will be obtaining "unspecified" or "extended" consent from participants, will you be recording individual participants' results with their personal records?*

- ○ Yes
- ◉ No

8.5.4 Will the results relating to specific participants be reported to anyone other than those participants?*

- ○ Yes
- ◉ No

Disseminating results

8.6.1 Is the research likely to reveal a significant risk to the health/wellbeing of persons other than the participant? *e.g. family members, colleagues, etc.**

- ○ Yes
- ◉ No

8.6.2 Is there a risk that the dissemination of results could cause harm of any kind to individual participants or to their communities?*

- ○ Yes
- ◉ No

8.6.3 How is it intended to disseminate the results of the research? *e.g. report, publication, thesis, etc.**

The results will be disseminated on the PhD thesis, relevant conference papers, and relevant journal papers.

8.6.4 Will the confidentiality of participants and their data be protected in the dissemination of research results?*

- ◉ Yes
- ○ No

8.6.4.1 Please explain how confidentiality of participants and their data will be protected in the dissemination of research results.*

The data will be presented anonymously. If there is pictures/videos of participant, the face of her/him will be blurred using computer application.

Attachments

Documentation

Your application is *nearly* complete. Please attach all the necessary documents below and accept the terms of submission on the next page.

To begin attaching items:

1. click **Add New Document**
2. place the cursor in the textbox and **type the name of the attachment** (as listed above)
3. click on the **green tick** to confirm the name
4. click on the **Soft copy icon** to open the browsing window and select a file
5. **press OK** to attach (and repeat the process as necessary).

10.1 **Attachments table** *(limit = 40MB per attachment)*

This question is not answered.

Based on your responses, you will need to attach the following items on this page as a minimum requirement:

l. Recruitment information/invitation(s)*
○ Attached ○ Not attached
This question is not answered.

m. Participant and Consent information (e.g. PICF on MQ letterhead)*
◉ Attached ○ Not attached

o. Statement (in consent information) regarding future use of data/biospecimens and the form(s) in which it will be shared (e.g. non-identifiable)*
◉ Attached ○ Not attached

10.2 Is there anything other than the items listed above that the applicants wish to attach to this application?
◉ Yes ○ No

10.2.1 Please explain the nature of these additional attachments.*

The experimental design is attached as well. This documents contains the information about the experimental design of this study in more details.

Signoff

Metadata

In order to leverage publically-funded research and sharing of results (where appropriate), the Australian National Data Service (ANDS) is encouraging the discovery, access and reuse of research data.

MQ is participating in this process by collecting metadata associated with projects completed by MQ researchers (i.e. descriptions of datasets collected through ethically-approved research projects).

This information will, ultimately, be forwarded to Research Data Australia (RDA), where researchers will be able to search for a description of your dataset among many other datasets currently available to be shared. **Click here** to see some examples of MQ datasets/collections already described in RDA.

For more information, please visit the Researcher toolkit page on the Research Office website.

11.1m **Do you consent to being contacted by MQ's ResearchOnline staff at the conclusion of your research to discuss sharing a description of your dataset (for potential use in future research projects)?***

Yes

Investigators

Prior to submission, students should consult with their supervisor(s).

Similarly, all other internal applicants are strongly encouraged to seek feedback on their application from the relevant Research Ethics Advisor prior to submission - please refer to the MQ Human Ethics website for your relevant contact.

11.1 **This question relates to all research personnel listed on this application.**

Please declare any conflict of interest that is likely to occur as a result of the signoff and ethical review process by indicating which of the following statements apply to you: *(please select all that apply)**

- ☐ I am a Head of Department or other signoff party
- ☐ I am/have been supervised by my Head of Department or other signoff party
- ☐ I am related to/in a spousal relationship with the relevant signoff party
- ☐ I am a member of a MQ HREC or Faculty Ethics Subcommittee
- ☐ I am related to/in a spousal relationship with a member of the relevant HREC/Subcommittee
- ☐ I am/have been a supervisor for a member of the relevant HREC/Subcommittee
- ☐ I am/have been supervised by a member of the relevant HREC/Subcommittee
- ☐ Other (not listed above)
- ☑ I foresee no potential conflict of interest in submitting this research proposal to MQ

11.2 Are there any further ethical considerations that you wish to raise? *(optional)*
○ Yes
◉ No

11.3 Signoff table *(please open, complete and save your own record)**

This question is not answered.

Once the signoff process has been completed by each investigator in the list above, please click the "Save" icon and proceed to the "Action" tab on the left-hand side of the screen for further options.

Office of the Deputy Vice-Chancellor
(Research)

Research Office
Research Hub, Building C5C East
Macquarie University
NSW 2109 Australia
T: +61 (2) 9850 4459
http://www.research.mq.edu.au/
ABN 90 952 801 237

MACQUARIE University
SYDNEY·AUSTRALIA

20 October 2016

Dear Associate Professor Vesilo

Reference No: 5201600506

Title: *Human activity recognition and fall detection using wearable sensors*

Thank you for submitting the above application for ethical and scientific review. Your application was considered by the Macquarie University Human Research Ethics Committee (HREC (Medical Sciences)).

I am pleased to advise that ethical and scientific approval has been granted for this project to be conducted at:

- Macquarie University

This research meets the requirements set out in the *National Statement on Ethical Conduct in Human Research* (2007 – Updated May 2015) (the *National Statement*).

Standard Conditions of Approval:

1. Continuing compliance with the requirements of the *National Statement,* which is available at the following website:

http://www.nhmrc.gov.au/book/national-statement-ethical-conduct-human-research

2. This approval is valid for five (5) years, subject to the submission of annual reports. Please submit your reports on the anniversary of the approval for this protocol.

3. All adverse events, including events which might affect the continued ethical and scientific acceptability of the project, must be reported to the HREC within 72 hours.

4. Proposed changes to the protocol and associated documents must be submitted to the Committee for approval before implementation.

It is the responsibility of the Chief investigator to retain a copy of all documentation related to this project and to forward a copy of this approval letter to all personnel listed on the project.

Should you have any queries regarding your project, please contact the Ethics Secretariat on 9850 4194 or by email ethics.secretariat@mq.edu.au

The HREC (Medical Sciences) Terms of Reference and Standard Operating Procedures are available from the Research Office website at:

http://www.research.mq.edu.au/for/researchers/how_to_obtain_ethics_approval/human_research_ethics

The HREC (Medical Sciences) wishes you every success in your research.

Yours sincerely

Professor Tony Eyers
Chair, Macquarie University Human Research Ethics Committee (Medical Sciences)

This HREC is constituted and operates in accordance with the National Health and Medical Research Council's (NHMRC) *National Statement on Ethical Conduct in Human Research* (2007) and the *CPMP/ICH Note for Guidance on Good Clinical Practice*.

Details of this approval are as follows:

Approval Date: 12 October 2016

The following documentation has been reviewed and approved by the HREC (Medical Sciences):

Documents reviewed	Version no.	Date
Macquarie University Ethics Application Form		Received 7/9/2016
Correspondence responding to the issues raised by the HREC (Medical Sciences)		Received 10/10/2016
Recruitment letter	1*	7/9/2016
MQ Participant Information and Consent Form (PICF) entitled	1.1*	10/10/2016
Experiment Design	1*	7/9/2016

***If the document has no version date listed one will be created for you. Please ensure the footer of these documents are updated to include this version date to ensure ongoing version control.**