Fall detection using a networked UWB radar system

Wenqiang Lai
B.Eng(*Hons*)

A Thesis submitted in fulfilment of requirements for the degree of
Master of Science
Applied Machine Learning
of Imperial College London

Department of Electrical and Electronic Engineering
Imperial College London
June 26, 2023

Abstract

Falls are the leading cause of death in the current ageing society. An automatic indoor fall detection method could accelerate the intervention after a fall, potentially leading to better treatment outcome. Existing technologies suffered from various constraints (e.g., high cost and low robustness toward environmental changes), for which these have not been widely deployed. This study comparatively evaluated 5 machine learning methods, namely SVM, RF, sCNN, ResNet18, KD-sCNN, based on data collected from a multistatic IR-UWB radar system sampling at a low rate of 10 Hz. The results revealed that SVM based on PCA extracted features and KD-sCNN that was boosted by the benchmark model ResNet18 via knowledge distillation were the most promising classifiers for the proposed system. Under standard strategy, SVM scored 98.9% and 100% test accuracy in multi-class and binary classification, while KD-sCNN achieved 98.3% in binary classification and 99.4% in multi-class scenario. Both classifiers were proved to be robust against domain shift between subjects and signal translation/rotation caused by radar displacement, due to which up to 4.2% accuracy reduction was observed for SVM and KD-sCNN. SVM and KD-sCNN occupied 1.25 and 0.141 MB of memory respectively, and time taken for one inference was only few milliseconds for both. Such low computational complexity and high prediction accuracy served as a great proof of the feasibility of the overall system proposed.

Acknowledgment

I would like to thank my co-supervisor Mr Bannon Alan for his continuous support throughout the year. Mr Hadjipanayi Charalambos, Mr Maowen Yin and Dr Adrien Rapeaux, who worked with me in this year, guided me at the beginning of the this study, when my knowledge about radar system was nearly zero, and provided precious advice for problems I struggled with. A special thank to Prof Timothy Constandinou, who has provided all the resources required for this study to be conducted. A special thanks to my friends at Imperial, whom I fought with against unprecedented challenges during this year.

Contents

Abstract **3**

Acknowledgment **5**

Contents **7**

List of Figures **9**

List of Tables **11**

Abbreviations **13**

Chapter 1. Introduction **15**
1.1 Background 15
1.2 Objectives 16

Chapter 2. Literature Review **19**
2.1 Existing Technologies for Fall Detection 19
2.1.1 Wearable Sensing-Based Technologies 19
2.1.2 Vision-Based Technologies 20
2.1.3 Radio-Frequency Radar-Based Technologies 21

Chapter 3. Data Acquisition **25**
3.1 Experimental Setup 25
3.1.1 Hardware 25
3.1.2 Laboratory Environment 27
3.1.3 Experimental Design 27
3.2 Data Processing 29
3.2.1 Dataset Overview 29
3.2.2 Signal Processing 29
3.2.3 Frequency Domain Analysis 33

Chapter 4. Methods **35**

4.1 Conventional Machine Learning Classifier . . . 35
4.1.1 Principal Component Analysis . . . 36
4.1.2 t-Distributed Stochastic Neighbor Embedding . . . 36
4.1.3 Support Vector Machine . . . 38
4.1.4 Random Forest . . . 39
4.2 Deep Learning Methods . . . 40
4.2.1 Convolutional Neural Network . . . 40
4.2.2 Residual Learning . . . 41
4.2.3 Knowledge Distillation . . . 42
4.2.4 Proposed Method: KD-sCNN . . . 43
4.3 Training Strategies . . . 44
4.4 Method Evaluation . . . 45

Chapter 5. Result and Discussion **47**
5.1 Visualisation of Extracted Features . . . 47
5.2 Results from Classifiers Trained on Standard Strategy . . . 49
5.3 Results from Classifiers Trained on Leave-One-Subject-Out Strategy . . . 51
5.4 Results from Classifiers Trained on Leave-One-Channel-Out Strategy . . . 53
5.5 Discussion . . . 55
5.5.1 General Performance of Evaluated Methods . . . 55
5.5.2 Feasibility of Evaluated Methods for Practical Use . . . 56

Chapter 6. Conclusion and Future Works **59**
6.1 Conclusion . . . 59
6.2 Limitation and Future Works . . . 60

Bibliography **61**

List of Figures

2.1 Frequency change over time of (a) frequency-modulated continuous wave, and (b) stepped-frequency continuous wave 22
2.2 Illustration of RF carrier signal pulses 23

3.1 A radar node in an adjustable mount. (left) A X4M05 radar mounted at the front. (right) A Raspberry Pi Zero W mounted at the rear. 25
3.2 A single frame recorded using X4M05 radar in an empty room. Each range bin is equivalent to around 6.43 mm. 26
3.3 (left) Fixed positions of radar nodes in the room. (right) Scaled layout of the room, where the orange circles, blue rounded rectangle, and dark grey rectangles represent radar nodes, air mattress and obstacles, respectively; subjects performed activities only in the light blue area, and the grey area was disregarded in the experiment. 27
3.4 Data structure of collected samples, where M and N represent the number of frames and bins, respectively. 30
3.5 (a) A raw data frame collected using radar 1 from Subject 9 performing a walk; (b) signal after background subtraction; (c) signal after SVD-based clutter removal; (d) signal after high-pass filtering. 31
3.6 A raw data sample collected using radar 1 from Subject 9 performing a walk: (a) raw data;(b) after background subtraction; (c) after SVD-based clutter removal; (d) after high-pass filtering. 32

4.1 A scree plot from [1]. The eigenvalues of 23 principal components were sorted in descending order and plotted as a scatter plot 37
4.2 An illustration of Random forest classifier 39
4.3 An illustration of convolution operation. With a input shape of (4,4), kernel size of (3, 3), unit step stride and no padding, the output feature map has size (2, 2) 40

4.4 Structure of the residual block in ResNet [2]. 41
4.5 ResNet18 architecture with the addition of an adaptive average pooling layer, which scales any input to the size of $(224, 224)$, removing the input dimension limitation. 42
4.6 The proposed KD-sCNN architecture is shown in the yellow block. 43

5.1 Column and row Scree plots generated by 2D-2D PCA. "Elbow" points clearly identifiable at 10^{th} column PC and 12^{th} row PC. 48
5.2 t-SNE visualisation of 240-D data on a 2-D space. 48
5.3 Confusion matrix generated from the predictions over test set given by SVM and RF trained by standard strategy. 49
5.4 Training and validation accuracy and loss curves, and confusion matrix from sCNN (top row), ResNet18 (middle row) and KD-sCNN (bottom row) over test set. 50
5.5 Cumulative confusion matrix over 10 trials from SVM (left) and RF (right) trained on leave-one-subject-out strategy. 51
5.6 Confusion matrices of all methods trained on leave-one-channel-out strategy. 54

List of Tables

3.1 Number of samples of each classes and each sub-class in the dataset 29

4.1 Definition of performance evaluation metrics 46

5.1 Performance of SVM and RF trained using standard strategy for 6 sub-classes classification 49

5.2 Performance of sCNN, ResNet18 and KD-sCNN trained using standard strategy for 6 sub-classes classification 50

5.3 Performance of SVM and RF trained using leave-one-subject-out strategy for 6 sub-classes classification 52

5.4 Test accuracy of sCNN, ResNet18 and KD-sCNN trained using leave-one-subject-out strategy. Data from Subject 4 and 5 were reserved as test set respectively. 52

5.5 Performance of all classifiers trained using leave-one-channel-out strategy for 6 sub-classes classification 53

5.6 Performance of all classifiers trained using leave-one-channel-out strategy for binary classification 53

Abbreviations

BS:	Background Subtraction
CNN:	Convolutional Neural Network
DoF:	Degree of Freedom
KNN:	K-Nearest Neighbour
KD:	Knowledge Distillation
PCA:	Principal Component Analysis
RF:	Radio Frequency
SNR:	Signal-to-Noise Ratio
STFT:	Short-Time Fourier Transform
SVD:	Singular Value Decomposition
SVM:	Support Vector Machine
UWB:	Ultra-wide band

Chapter 1

Introduction

1.1 Background

THE demand for better health care for elderly people increases with the ageing of population. Falling is a common cause of injury among the elderly. In the United States, around 36 millions falls were reported by individuals aged 65 years or older in the 2018, resulting in more than 30 thousands deaths [3]. The rate of death caused by falling increased by 31% from 2007 to 2016 [4]. According to [5], early intervention after a fall could improve the outcome of treatment. Therefore, a real-time automatic fall detection system could be a potential solution for improving health care provision, especially for the elderly living alone. A variety of technologies have been developed for fall detection, which can be categorised according to the type of sensor used, such as cameras, wearable sensors, and radars [6–9]. Vision-based technologies generally rely on human body parts tracking, whereas wearable sensors, such as accelerometer, could detect abrupt body parts acceleration caused by falls. The emergence of depth camera, such as Kinect, addressed the problems of traditional vision-based systems using 2D cameras, such as vulnerability to lighting conditions and privacy concerns [10]. However, the problem of obstacle occlusion persists, and the relatively high installation cost hindered such systems in deploying in a domestic environment. Wearable sensor-based methods require strict user compliance (e.g., actively wearing or charging the device), which is not friendly to

elderly users, especially those suffering from dementia. Recently, radar-based systems are considered emerging alternatives due to their ability of measuring body movement reliably and unobtrusively [10]. In this study, a multistatic networked radar system named Tiresias [11], comprising of several Novelda X4M05 radar modules (up to 4 modules), will be used to detect falling events in a domestic environment. X4M05 radar is an impulse radio ultra-wideband (IR-UWB) radar operating at sub 10 GHz, allowing high range resolution, penetration through obstacles (e.g., walls), and target detection up to 9.9 meters away [12, 13]. In this study, the radars will be placed at fixed locations to allow maximum coverage.

The rest of this report will be organised as follows. Chapter 2 will introduce the working principles of IR-UWB radars and related works using similar technologies. Chapter 3 will describe the experimental setup for data acquisition and data pre-processing techniques. Chapter 4 will detail the feature extraction techniques and the proposed machine learning methods. A comprehensive result analysis will be given in Chapter 5. Lastly, Chapter 6 will summarise key findings and conclude with future work recommendations.

1.2 Objectives

This study will explore the feasibility of automatic fall detection using the multistatic radar system Tiresias with a sampling frequency of 10 Hz. Towards the end of study, a method capable of effectively distinguishing falling events from false alarms (e.g., sitting or walking) will be proposed. To achieve the final goal, the following objectives must be met:

- Recruit at least 10 experiment subjects, from whom a dataset comprising of two classes, Fall and Non-Fall, will be generated. The Fall class will contain three sub-classes, representing different types of falling; to maintain the balance of the dataset, the Non-Fall class will also comprise three sub-classes, corresponding to other activities that might cause false alarms. The balance between subject will also be kept, in orther words, each participant will contribute similar amount (mean no. of samples

per subject ±10%) of samples to the dataset.

- Filter out any outliers from the dataset and apply pre-processing techniques to mitigate noises.
- Extract features from the dataset using unsupervised methods, and use classic classifiers, e.g., SVM and Random Forest, to perform fall detection. Methods with an accuracy of above 90% on the test set would be considered successful.
- A number of deep learning methods, e.g., CNN-based neural networks, will be implemented in comparison to the classic methods. Same successful criteria apply for deep learning methods.

The generalisation ability of models will be evaluated via the leave-one-subject-out strategy. The effect of radar positional shift will also be explored to test the robustness of proposed methods.

Chapter 2

Literature Review

2.1 Existing Technologies for Fall Detection

NUMEROUS technologies have developed for the detection of falling events. According to the types of sensor used, this chapter will introduce the working principles of various technologies, and review relevant studies, while give a stronger focus on the radar-based solutions.

2.1.1 Wearable Sensing-Based Technologies

In the past decades, wearable devices have been extensively used in the field of human activity recognition [14]. Inertial sensors, such as accelerometer and gyroscope, were mounted on human body to capture the signal generated by the movement. Since the falling of human body causes abrupt changes in signals, threshold-based detection methods have been proven to provide satisfactory classification accuracy. In [15], two tri-axial accelerometers were fitted on the trunk and thigh of participants, and by setting an upper and a lower threshold for the signals generated from each sensor, the proposed algorithm achieved 100% classification accuracy. Later, Li *et al.* [16] proposed a system comprising accelerometers and gyroscopes. In addition to the accelerometer-based posture recognition, gyroscopes were used to determine whether the transition between recognised postures was intentional (e.g., fall would be unintentional transition to a lying position). The

algorithm was similarly based on threshold calculated from recorded data, and the binary classification accuracy was 92%.

Despite the high classification accuracy, wearable device-based solutions are inconvenient for prolonged use. An inherent constraint of wearable systems is that the users are required to wear them throughout the day. Any system based on wired connection would be strongly disadvantaged due to the added obtrusiveness. For compact wearable systems, limited battery power would bring inconvenience to users, especially to the elderly. Also, inertial sensors require regular calibration to mitigate the error accumulated during the usage [17], which further increased inconvenience.

2.1.2 Vision-Based Technologies

Computer vision techniques have been widely studies and applied in a variety of domains due to the rapid development of hardware and software in recent years [18]. As a result, computer vision-based solutions formed another major category in the area of fall detection. Vision-based methods could be broadly divided into two sub-categories, 2D and 3D, depending on the type of features used. In 2D vision-bases systems, a single RGB camera was generally used to record video data, from which, features as bounding box, silhouette and the center of gravity (COG) of human body were extracted by applying computer vision techniques (e.g., image segmentation, edge detection, etc.) [19–21]. In 3D systems, multiple RGB cameras or a single depth camera were used to collect depth information, from which, features as 3D bounding box, 3D silhouette and body joint position could be extracted. In earlier studies, threshold-based classification algorithms were generally applied to the extracted features. In [22], the height/weight (HW) ratio of 2D bounding box was computed and compared with a subject-specific threshold value. The classification accuracy was 68% when only HW ratio was considered in computing the threshold value, while the accuracy increased to around 80% when extra information were taken into account (e.g., health record). By thresholding the distance between 3D body centroid and the floor plane, Diraco *et al.* [23] achieved approximately 80% accuracy. Compared to wearable systems, vision-based approaches were less obtrusive, as camera

could be conveniently installed at any location. However, vision-based systems were vulnerable to occlusions, which are likely to happen in a domestic environment. Zhang *et al.* [24] evaluated four fall detection methods based on a single depth camera, and revealed that three out of four evaluated methods were invalid on occluded falls as the accuracy fell below 50%. Also, vision-based solutions generally have high hardware requirement, especially for a real-time detection, which increases the deployment cost of such systems. Another fundamental issue was that camera surveillance might raise privacy concerns.

2.1.3 Radio-Frequency Radar-Based Technologies

Fall detection approaches based on radar technologies could overcome the problems encountered by aforementioned systems. In the existing literature, radio-frequency (RF) radars were generally used for remote monitoring.

RF Radar Fundamentals

RF radars could be used to determine the distance to a target, and they work by emitting radio waves with a frequency of a few GHz, which will reflect back when hit on an obstacle. RF radars could be broadly categorised into continuous wave (CW) and pulse wave radar. The simplest form of CW radar uses unmodulated CW wave of fixed frequency to measure the velocity of a moving target. Due to the Doppler effect, the movement of a target causes changes in the frequency of reflected signal, which is known as Doppler shift f_d, and the velocity of the target v could be calculated by Equation 2.1, where f is the frequency of transmitted radio wave [25, p. 274].

$$v = \frac{cf_d}{2f} \tag{2.1}$$

Frequency-modulated CW (FMCW) and stepped-frequency CW (SFCW) radars were the popular CW-based radar as they could provide range information of a target. These two radar are similar as they emit radio waves of frequency varying over time. FMCW linearly varies frequency of emitted waves within a period of time, while SFCW varies the frequency by discrete steps (Figure 2.1). By computing the frequency difference between the received wave and the wave about to emit at a particular time, the time-of-flight of

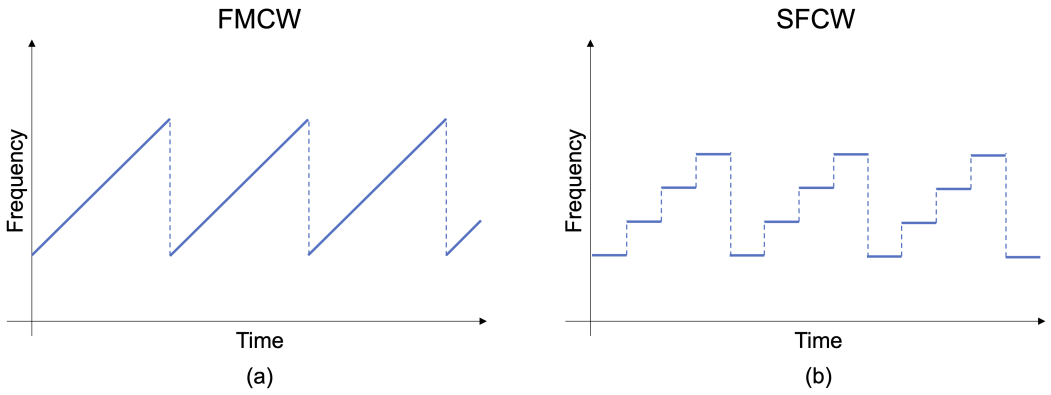

Figure 2.1: Frequency change over time of (a) frequency-modulated continuous wave, and (b) stepped-frequency continuous wave

the wave could be obtained. The range R to a target would be the time-of-flight ΔT multiplied by the speed of light c divided by 2 (Equation 2.2) [25, p. 4].

$$R = \frac{c\Delta T}{2} \tag{2.2}$$

Pulse wave-based radars emit distinct pulses of RF waves, as shown in Figure 2.2,towards the target, and use the time-of-flight of reflected pulse waves to provide range information. Impulse radio ultra-wide band (IR-UWB) radar is a common type of pulse radar, which allows high spatial resolution due to its large bandwidth ($>$ 500 MHz) [26]. The smallest range increment is referred to as range bin, which could be calculated using Equation 2.3, where $f_{receiver}$ corresponds to the sampling frequency of the radar receiver. The measurements of range happen in a small-scale time dimension, also known as fast-time. A collection of fast-time measurements across a time dimension of larger scale, also known as slow-time, is referred to as radar range profile, which is a common representation of raw radar data.

$$\Delta R = \frac{c}{2f_{receiver}} \tag{2.3}$$

Compared with the CW counterparts, IR-UWB exhibited higher obstacle penetration ability and robustness to clutter in vital sign measurements [27].

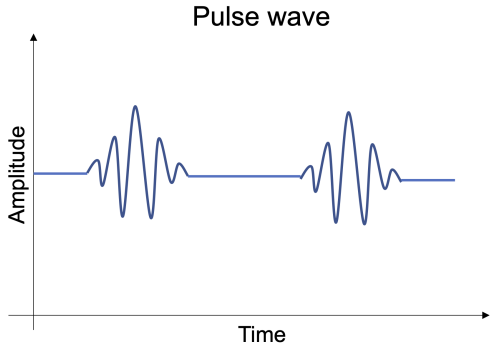

Figure 2.2: Illustration of RF carrier signal pulses

RF Radars for Fall Detection

Various RF radars have been used for fall detection. Wu *et al.* [9] implemented a fall detection system based on a single CW radar with a sampling frequency of 1 kHz. The proposed method was purely based on the features extracted from the 4-second spectrograms generated using STFT. The spectrograms were processed using standard image processing techniques, since they could be considered as grey-scale images. The proposed classification algorithm based on revelance vector machine (RVM), which is a probabilistic version of support vector machine (SVM), resulted in a binary classification accuracy of 98.75. However, the sample size was only 80, and the classes were imbalanced, where only 20% of samples were Fall samples. Jokanovic *et al.* proposed a Principal Component Anaylisis (PCA)-based approach using a monostatic CW radar sampling at 1 kHz. STFT was performed and the resulting spectrograms were projected to a lower dimension space formed by the largest principal components. The minimum Euclidean distance in the low-dimensional space between the training samples and test samples was used for classification. This PCA-based approach achieved 90% binary classification accuracy on a test set consisting of 30 samples. Using narrowband pulse wave radar, Wu *et al.* proposed a system based on Hiden Markov Model (HMM), which achieved 96.7% binary classification on both features extracted using STFT and matching pursuit decomposition (MPD). More recently, deep learning methods have been applied for radar-based fall detection system. Yoshino *et al.* [28] proposed a CNN-based approach that automatically extract hidden

features from the spectrogram, which scored 93.4% in binary classification, and 91.3% in the classification of 4 sub-classes (2 fall and 2 non-fall classes). Maitre *et al.* proposed a system based on the CNN-LSTM architecture, where the CNN part serves as feature extractor and the short-term memory (LSTM), a type of recurrent neural network with addition of memory cells to mitigate gradient vanishing problem induced by long-term dependencies in signals [29], acts as classifier. Three UWB radars were fixed at distant locations in a domestic environment, and the collected 2-dimensional raw data was processed into 1-dimensional vector normalised between 0 and 1. The system was evaluated using the so-called leave-one-subject-out, where data from all subjects except one was used for training and test the classifier on the left out samples; this process is repeated until all subjects has been left out exactly once, the the test accuracy across subjects is averaged. This system scored 88.1%, 91.5% and 82.6% binary classification accuracy on falls performed in the 3 positions. Excluding the radars distant away from the position of falls, the accuracy increased to 98.5%, 96.8% and 95.5%.

Chapter 3

Data Acquisition

3.1 Experimental Setup

IN order to validate whether networked IR-UWB radars could be used to detect falls, an experiment has been conducted to generate a dataset.

3.1.1 Hardware

Two Novelda X4M03 radars networked via wireless connection have been used for this project. The X4M03 radar is ideal for the purpose of indoor real-time fall detection due to its compact size and low power consumption. The default carrier frequency $f_{carrier}$ of radar was set to 7.29 GHz to comply with ETSI regulations [12].

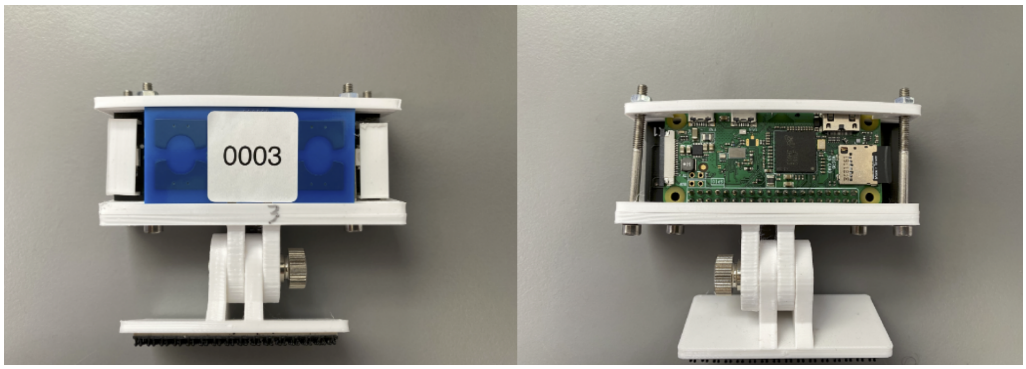

Figure 3.1: A radar node in an adjustable mount. (left) A X4M05 radar mounted at the front. (right) A Raspberry Pi Zero W mounted at the rear.

Each radar was stacked with several PCB boards and a Raspberry Pi Zero W, and mounted in a 3D printed adjustable stand (Figure 3.1). A radar node software was deployed on the Raspberry Pi, and it was responsible for receiving radar data and networking via the local Wi-Fi provided by a LAN router, through which the radar data were transmitted to a host device running a supervisor software. Radar data were stored in the form of frame, which is a vector containing amplitude values at each range bin. According to [12], the $f_{receiver}$ of X4M05 is 23.328 GHz, which yields a range resolution of 6.43 mm (Equation 2.3). Figure 3.2 illustrated one frame of radar data recorded in an empty room, which consisted of 934 range bins, implying a maximum detectable range of around 6 m. Note that large amplitudes were shown in the first 100 bins due to the cross-talk between transmitter and receiver, in other words, receiver directly captured the emitted signals, rather than capturing their reflections.

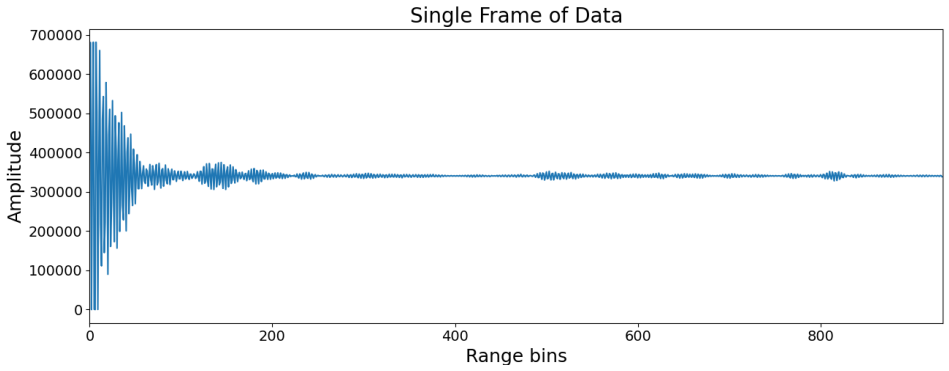

Figure 3.2: A single frame recorded using X4M05 radar in an empty room. Each range bin is equivalent to around 6.43 mm.

Due to the Wi-Fi bandwidth constraint, the sampling frequency across slow-time f_s was set to 10 Hz. A laptop (MacBook Pro 2021) was used to run a supervisor software and a self-developed GUI for data collection. The supervisor software was responsible for radar configuration, data transmission, as well as data alignment, because the incoming frames of data were not chronologically ordered [11]. The built-in camera of the laptop was used to record video data, which was used to validate the data at the end of experiment.

3.1.2 Laboratory Environment

The experiment was conducted in an indoor environment, since the fall detection methods mainly target elderly people in domestic environments. Figure 3.3 shows the internal environment of the laboratory. Irrelevant objects were removed to prevent subjects from injuries and mitigate clutter causes by non-stationary objects. While every effort was made, there might still be sources of noise present in the room (e.g., PC cooling fan), however, the impacts were negligible. An air mattress was placed in the room to mitigate risk of injury when falling events were simulated.

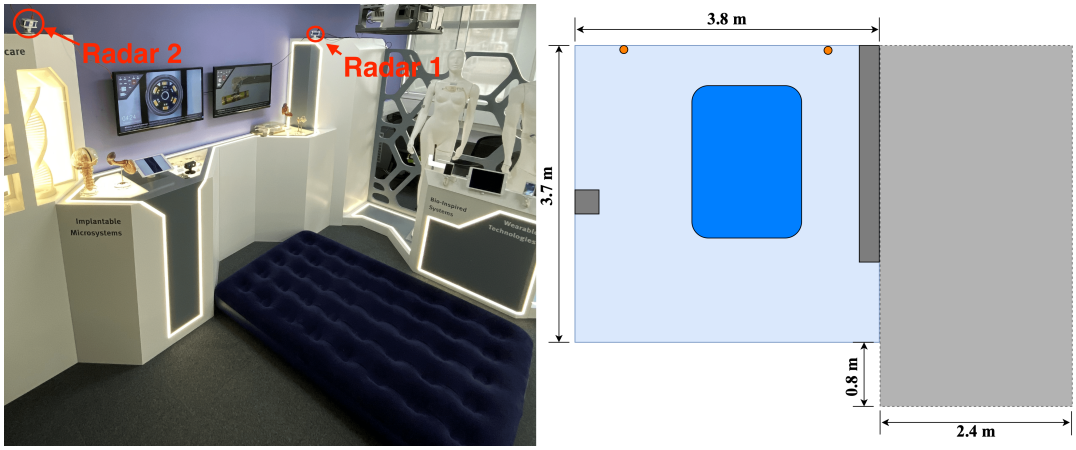

Figure 3.3: (left) Fixed positions of radar nodes in the room. (right) Scaled layout of the room, where the orange circles, blue rounded rectangle, and dark grey rectangles represent radar nodes, air mattress and obstacles, respectively; subjects performed activities only in the light blue area, and the grey area was disregarded in the experiment.

3.1.3 Experimental Design

Two radars were placed at two fixed locations at 2.1 meters above the ground with a distance of 2.5 meters between the radars. In such a configuration, radars were capable of detecting movements perpendicular to the floor plane. Since the experiment require subjects to simulate falling events, 10 healthy subjects aged between 22-26 have been recruited. Subjects were asked to perform 3 actions for each of the 2 classes, *Fall* and *Non-Fall*, where each action was repeated for 30 times. Actions were designed based on the reviewed literature. Three types of fall were simulated, namely *Stand Fall*, *Sit Fall* and *Walk Fall*. For each type of fall, subjects were required to fall forward, toward left

and toward right (10 falls for each):

- **Stand Fall**. Subjects stand for approximately 1s and then fall down.
- **Sit Fall**. Subjects stand up from sitting and then fall down; this was to simulate the dizziness and imbalance caused by postural hypotension, a leading cause of falls among elderly people [30].
- **Walk Fall**. Subject walk at normal speed toward the air mattress and then fall down; this was to simulate falls caused by tripping, which was reported to cause most falls among the elderly people in [31].

After the fall, subject remained still laying on the mattress until a notice was given. For the *Non-Fall* class, 3 daily activities representing 3 sub-classes were performed, and are referred to as *Stand Non-Fall*, *Sit Non-Fall* and *Walk Non-fall*:

- **Stand Non-Fall**. Subjects stand for approximately 1s and then simulate picking an object from the ground.
- **Sit Non-Fall**. Subjects stand for approximately 1s and then sit on a chair; the aim is to confound the classifier as the sitting is a rapid postural change similar to falling.
- **Walk Non-Fall**. Subjects simply walk from one point to another at a normal walking speed.

Note that subjects conducted experiment on different dates, however, within similar time slots (all between 2PM to 6PM) to avoid potential confounding factors. At the end of experiment of each subject, the radar data were visualised and validated against video data to assign labels to data samples and eliminate data samples that did not comply with the aforementioned requirement for each sub-class. After this process, the video data were immediately eliminated to protect the privacy of subjects.

3.2 Data Processing

3.2.1 Dataset Overview

Table 3.1: Number of samples of each classes and each sub-class in the dataset

Class	Sub-class	No. of samples
Fall		898
	Stand	299
	Sit	300
	Walk	299
Non-Fall		897
	Stand	298
	Sit	300
	Walk	299

A total of 1795 samples were collected, and the number of samples of each class and each sub-class is summarised in Table 3.1. The number of samples is similar between class and classes, which prevent the problems caused by class imbalance [32]. Note that each subject contributed similarly, providing extra insurance to the balance of dataset. Each sample consists of 40 frames (as illustrated in Figure 3.2), and with a f_s of 10 Hz, this is equivalent to a window size of 4 seconds, which is the average time for a falling event to complete according to [33]. Thus, the data samples from each radar were stored as MxN matrices, where $M = 40$ and $N = 934$ (Figure 3.4). The overall shape of the dataset would be (1795,2,40,934) where second dimension corresponds to the number of channels (radars).

3.2.2 Signal Processing

The raw data were hardly interpretable as the useful information was concealed by the background noise. As shown in a raw data sample collected from Subject 9 performing a walk toward radar 1, the moving target was not distinguishable from the noises and clutters (Figure 3.6(a)). Therefore, some processing was required in order to increase the signal-to-noise ratio (SNR). The main sources of noise were the pulse echoes from stationary objects and the antenna cross-talk. Since these noises were almost constant in each frame of signal, several conventional signal processing techniques capable of separating moving target from

stationary background were applied and evaluated. By convention, the amplitudes were converted to power. Afterwards, a frame-wise standardisation was applied according to Equation 3.1, where the frame mean $\bar{\mathbf{x}}$ was subtracted by each bin value x_i, which then was divided the frame standard deviation σ_x.

$$y_i = \frac{x_i - \bar{\mathbf{x}}}{\sigma_x} \tag{3.1}$$

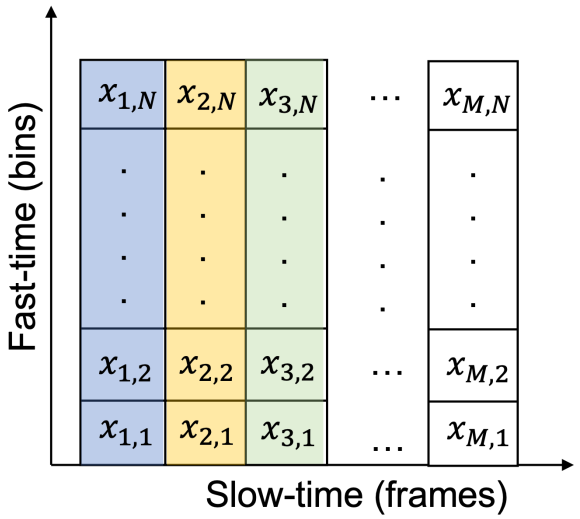

Figure 3.4: Data structure of collected samples, where M and N represent the number of frames and bins, respectively.

Background Subtraction

Background subtraction (BS) techniques were commonly employed to separate the foreground from the noisy background due to their simplicity [34]. A 200-frame sequence of data $\mathbf{X_{empty}}$ was recorded in an empty room, and averaged across the slot-time axis to give a reference frame $\mathbf{x}_{empty}$. By subtracting the $\mathbf{x}_{empty}$, the DC component in the signal should be eliminated or attenuated. However, as shown in Figure 3.5(b) where a single frame collected from a subject performing a walk, BS failed to remove the large peaks near range bin 800 (around 5 m from the radar), which was obviously caused by clutters. This resulted in a partial separation of the moving target from the background as depicted by the range profile from a walking subject (Figure 3.6(b)), where a blurred boundary was

formed along the trajectory of the moving target (dashed black line). Ideally, only the movement of a target should result in large power peaks in the range profile. However, with BS, the peaks coincided with the location of target only at at around time $t = 0$, after which, the target was concealed by large and consistent peaks in the region between 3 and 4.5 meters. The consistent large peaks could be induced by the difference in the environment when the empty-room and walking data were collected (e.g., addition or removal of an obstacle), which caused variation in amplitudes.

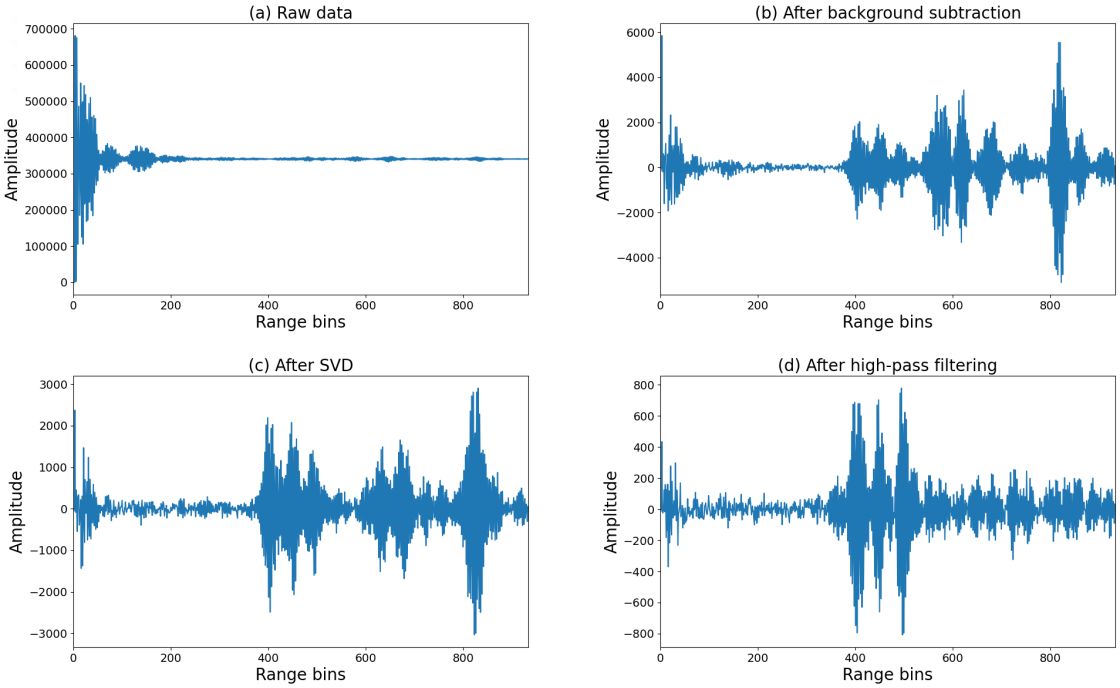

Figure 3.5: (a) A raw data frame collected using radar 1 from Subject 9 performing a walk; (b) signal after background subtraction; (c) signal after SVD-based clutter removal; (d) signal after high-pass filtering.

Singular Value Decomposition

Singular Value Decomposition (SVD) could factorise the data matrix $\mathbf{X}$ (MxN) in the following form:

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathbf{T}} \tag{3.2}$$

where $\mathbf{U}$ ($M \times M$) and $\mathbf{V}$ ($N \times N$) are real unitary matrices since the data matrix $\mathbf{X}$ is real. The diagonal matrix $\mathbf{\Sigma}$ ($M \times N$) contains r non-zero singular values, where r equals

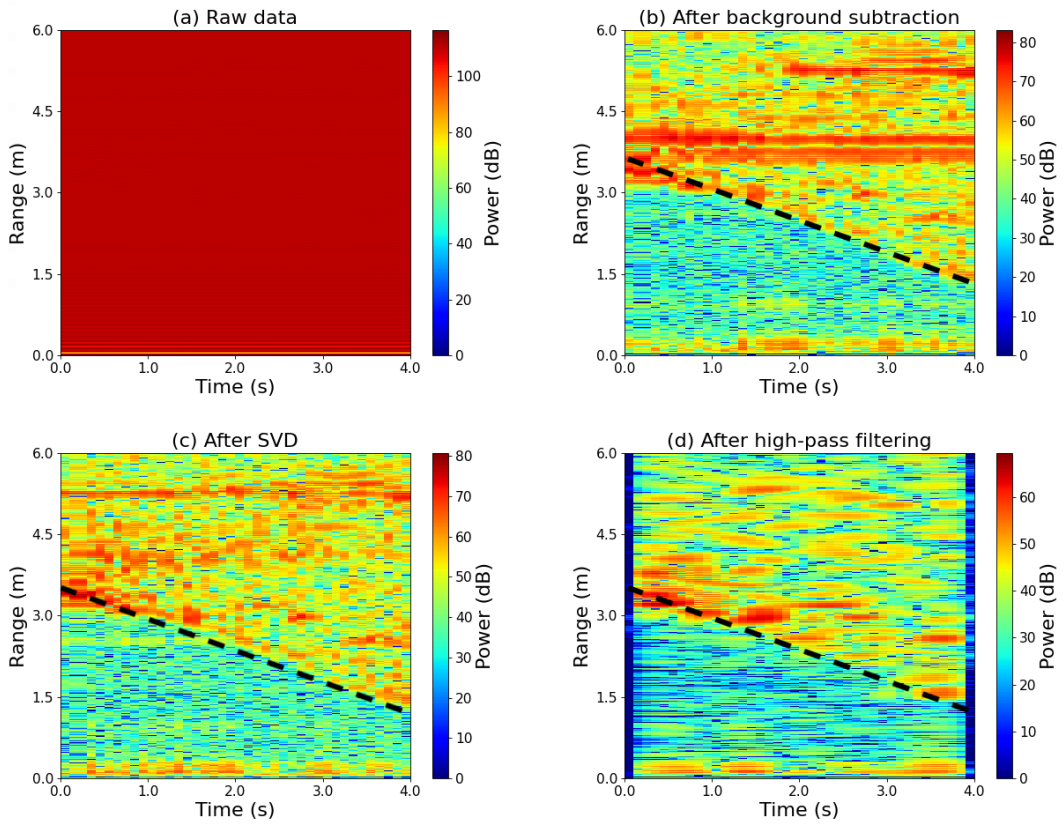

Figure 3.6: A raw data sample collected using radar 1 from Subject 9 performing a walk: (a) raw data;(b) after background subtraction; (c) after SVD-based clutter removal; (d) after high-pass filtering.

to the rank of $\mathbf{X}$. According to [35], the noise and clutter mainly resided in the subspace of the largest singular value. Thus, eliminating the largest singular value subspace during the reconstruction of $\mathbf{X}$ should remove the noise and clutters. As seen in Figure 3.5(c), SVD-based method the amplitudes of clutters were attenuated, approaching the amplitude of actual signal. In Figure 3.6(c), SVD-based method created a clear boundary coinciding with the walking trajectory. However, it was not possible to accurately localise the target, since the amplitude generated by the actual movement was close to the amplitude of clutters.

High-Pass Filtering

Following the same idea of suppressing low frequency components in the signal, a 4^{th}-order Butterworth high-pass filter was applied. Since our sampling rate was only 10 Hz, a

cut-off frequency of 4 Hz was used. With high-pass filtering, the clutters were effectively suppressed, significantly improving the SNR (Figure 3.5). Compared to other methods, the appearance of ghost target was greatly suppressed, and several peaks representing the true target were clearly visible along the trajectory (Figure 3.6(d)).

3.2.3 Frequency Domain Analysis

The aforementioned techniques were applied directly to the raw data samples, which were in the time domain. In the existing literature, the Short-Time Fourier Transform (STFT) was generally adopted, since the resulting time-frequency representation of radar signal yields more useful information than the time-distance representation. However, performing STFT on the walking data resulted in a spectrogram with very limited information because of the low sampling frequency. Also, to avoid aliasing, a system with a f_s of 10 Hz could only detect a Doppler shift $f_d < \frac{f_s}{2}$, according to Nyquist–Shannon sampling theorem. The constraint on the f_d yielded a maximum detectable speed, which is calculated to be 0.1 ms^{-1} (Equation 2.1). Obviously, such a low f_s was insufficient to measure the falling speed. As such, this study will solely focus on the time-distance representation of data.

Chapter 4

Methods

4.1 Conventional Machine Learning Classifier

FOR this study, two conventional classifiers, namely Support Vector Machine (SVM) and Random Forest (RF) were evaluated. Generally, task-specific features with high descriptive power are extracted from raw data before feeding into the classifiers. However, the manually selecting and extracting features require human intervention, which could unintentionally introduce artefacts. Thus, this study will directly use the raw data as input to the classifiers. Since the data used in this study were in the format of 2-channel 2D matrices, they could be considered as 2-channel images. Both SVM and RF have been successfully applied to high-dimensional tasks, including image classification [36]. In image classification tasks, SVM and RF consider each pixel as a feature, and an image of size (C,M,N), where C is the number of channels, will result in a feature vector of length $C \times M \times N$. Although SVM and RF have been proved to perform well with high-dimensional data (e.g., $p >> n$, where p is the number of features and n is the number of samples) [37, 38], applying dimensionality reduction technique beforesuch as Principal Component Analysis (PCA), feeding feature vector into the classifiers could improve the their computational efficiency, as well as reducing the risk of over-fitting.

4.1.1 Principal Component Analysis

PCA is a classic unsupervised method for down-sampling and feature extraction. PCA could effectively reduce the dimension of data by computing all the orthogonal bases, known as principal components, and constructing an approximation of the original data using only the principal components with top k-th largest variances. Therefore, applying PCA will reduce the dimension of data while retaining useful information. Two-directional two-dimensional (2D-2D) PCA, a PCA variant proposed by [39], was used in this study due to its effectiveness in dealing with 2-dimensional data. Considering a dataset consisting of N range profiles $\mathbf{R}_{m\times n}$, the 2D-2D works by finding the optimal projection matrices $\mathbf{X}_{m\times d}$ and $\mathbf{Z}_{q\times n}$ along the rows and columns, respectively, where d and q are the optimal number of principal components. By projecting the original data $\mathbf{R}$ on $\mathbf{X}$ and $\mathbf{Z}$, a feature matrix $\mathbf{C}_{d\times q}$ with reduced dimensionality could be obtained:

$$\mathbf{C} = \mathbf{Z}^T\mathbf{R}\mathbf{X} \tag{4.1}$$

The number of optimal principal components could be selected by analysing the Scree plot, which is a scatter plot of eigenvalues associated to each principal component in a descending order. A conventional approach is to find the so-called "elbow" of the scree plot, which is the point at which the rate of reduction attenuates (e.g., the 7-th principal components in Figure 4.1).

4.1.2 t-Distributed Stochastic Neighbor Embedding

It is often tricky to visualise data of dimensions higher than 3. Reducing the dimensionality to 2 or 3 using PCA usually result in crowded and inseparable data points when applied for 2 or 3D visualisation, since PCA performs the computation of variance globally (all points were considered), and only the first 2 or 3 principal components can only explain a small portion of variance of the entire dataset. In contrast, t-distributed Stochastic Neighbor Embedding (t-SNE), an unsupervised method proposed by [40], can overcome the limitation of PCA [41]. t-SNE works by minimising the Kullback-Leibler (KL) divergence

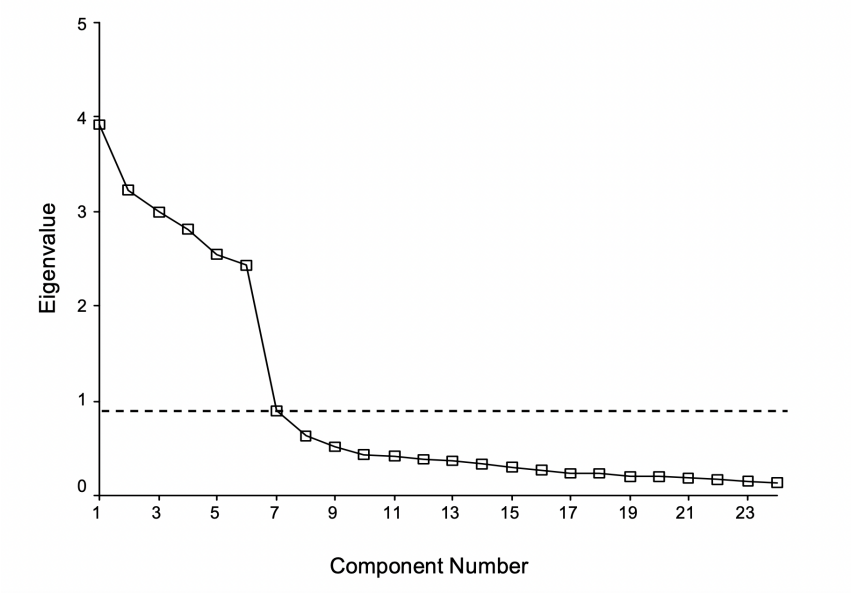

Figure 4.1: A scree plot from [1]. The eigenvalues of 23 principal components were sorted in descending order and plotted as a scatter plot

C between two joint probabilities q_{ij} and p_{ij}:

$$C = KL(P||Q) = \sum_i \sum_j p_{ij} log \frac{p_{ij}}{q_{ij}} \tag{4.2}$$

The joint probabilities p_{ij} represents the measures the affinity between points x_i and x_j in the original input space, which can be converted from the Euclidean distance between points i and j by means of Gaussian distribution. Considering a low-dimensional embedding $\mathbf{Y} = \{\mathbf{y_1}, \mathbf{y_2}, ..., \mathbf{y_n}\} \subset \mathbb{R}^d$, q_{ij} measures the affinity between points in $\mathbf{Y}$ using 1-DoF Student-t distribution. The gradient $\frac{\partial C}{\partial y_i}$ is given by:

$$\frac{\partial C}{\partial y_i} = 4 \sum_{j \neq i} (p_{ij} - q_{ij}) \frac{d_{ij}}{1 + d_{ij}^2} \tag{4.3}$$

where d_{ij} represents the distance between two points y_i and y_j. By gradient descent with momentum, $\mathbf{Y}$ is updated according to the following rule:

$$\mathbf{Y}^{(t+1)} = \mathbf{Y}^{(t)} + s \frac{\partial C}{\partial \mathbf{Y}} + \alpha(\mathbf{Y^{(t)}} + \mathbf{Y^{(t-1)}}) \tag{4.4}$$

where s and α are the step size and momentum respectively. Compared to PCA, dissimilar

points will result in large pairwise distance, and vice-versa for similar points.

4.1.3 Support Vector Machine

Before the rise of neural network-based methods, SVM was considered as the best performing supervised learning method for classification tasks [42]. SVM works by computing a set of support vectors using the data points, which forms a binary classification boundary that separate the data points. For hard-margin SVM, the error tolerance is zero, in other words, the data has to be separable. In contrast, soft-margin SVM allows some violations (data points crossing the boundary) and introduces a parameter C in the optimisation problem [43]:

$$\begin{aligned} \min \quad & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{N}\xi_i \\ \text{s.t.} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned} \tag{4.5}$$

where $\sum_{i=1}^{N}\xi_i$ is the total number of violations. The parameter C controls whether to prioritise the minimisation of the first term (maximal margin) or the second term (less violations). The smaller the C, the larger the margin and thus more tolerance to violations, vice-versa for a larger C. To deal with classification tasks with non-linear data, SVM makes use of kernel functions $K(\mathbf{x}, \mathbf{x}')$ to project the data points to a high-dimensional space where the they can be linearly separated. For this study, radial basis function (RBF), a popular kernel for SVM, was chosen, and it has the following form [44]:

$$K(\mathbf{x}, \mathbf{x}') = exp(-\gamma||\mathbf{x} - \mathbf{x}'||^2) \tag{4.6}$$

where the parameter γ controls the flexibility of classification boundary. With large γ, the hyperplane is more curved, leading to small number of violations, while with small γ, the hyperplane tends to be a linear boundary. Since a multi-class scenario was designed to evaluate the classifiers, this study employed a One-vs-One (OvO) training strategy. For N-class classification task, $N * (N-1)/2$ binary classifiers were trained, where each classifier is capable to classify a specific binary subset. A grid search method was employed

to find the optimum combination of C and γ [44].

4.1.4 Random Forest

RF has been a very successful conventional supervised learning method. The basic building block of RF is decision tree, which is built from top (root node) to bottom (leaf nodes). In a decision tree, the input samples are fed into the root node, and in all nodes except the leaf nodes, a binary test that maximised the information gain ΔH, which is the difference between the entropy of a given set $H(D)$ and the entropy of the subset $H(D_i)$ split from the parent set:

$$\Delta H = -\sum_{i} \frac{|D_i|}{D} H(D_i) \tag{4.7}$$

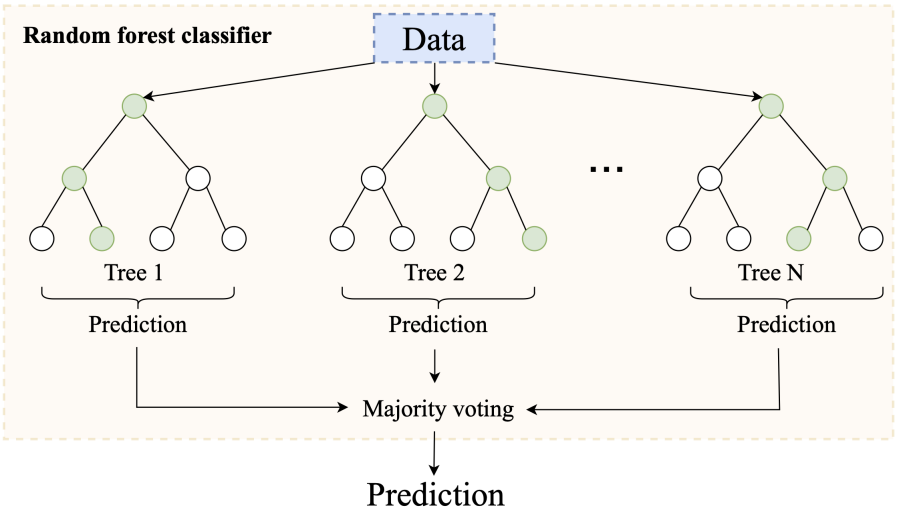

Figure 4.2: An illustration of Random forest classifier

The training stage involves the calculation of probabilities P that instance I arriving at leaf node l of tree t belongs to class c, $\forall l \in L, t \in T$, where L is the set of all leaf nodes and T is is the set of all trees. RF employs bagging method that randomly samples from the dataset with replacement to generate N subsets for each of N decision trees. A majority voting is performed on the predictions given by all decision tree, resulting in a final prediction. Likewise, RF require parameter tuning for an optimal performance. However, the number of parameters to be optimised are far more than SVM (e.g., number

of tree, max depth, max split, etc.). A grid search was also performed to find optimal parameters.

4.2 Deep Learning Methods

4.2.1 Convolutional Neural Network

CNN has gained a great attention since the success of AlexNet in image classification [45]. In contrast to conventional methods, CNN could be directly applied to raw data, from which the set of kernels will automatically learn the hidden features by convolution with the the data in its receptive field (Figure 4.3). The weights in the kernels are updated through the backpropagation algorithm [46]. A typical CNN-based architecture would stack several

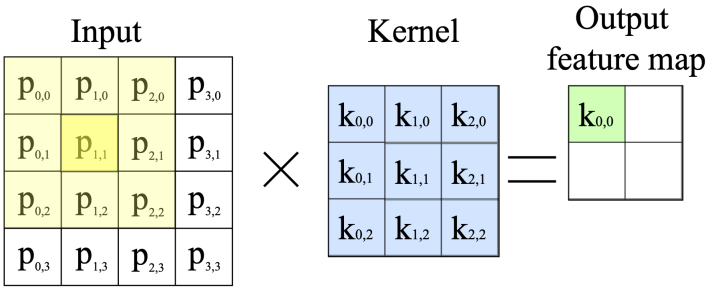

Figure 4.3: An illustration of convolution operation. With a input shape of (4,4), kernel size of (3, 3), unit step stride and no padding, the output feature map has size (2, 2)

CNN layers with interleaving pooling layer, which down-sample the input data, to produce robust features. The kernel size, or receptive field of a filter, is an important hyper-parameter. A large kernel size would capture more global information at the cost of high complexity. Small kernel size require substantially less computations, and by stacking multiple layers, the global information could be recovered. However, simply increasing the number of layers is prone degradation. With deep neural networks, the gradients are subject to vanishing/exploding problem when propagating to the bottom layer, leading to undesirable weight updates during backpropagation. Nummerous techniques have been introduced to mitigate the problem, such as using non-saturating activation function (e.g., Rectified Linear Unit) and Batch Normalisation (BN). For instance, Sigmoid is a saturating

activation function, whose gradient becomes flat for large input. ReLU takes the form of $f(x) = max(0, x)$, whose gradient is either 0 for $x < 0$ or 1 for $x > 0$; for $x = 0$, ReLU is not differentiable, and the gradient is set to 0 by convention. According to [47], normalising the input to each hidden layer could prevent the amplification impact from previous layers, while accelerating the training process. Despite the abundance of proposed solutions, the problem of degradation persists in deep neural networks. Dalduzzi *et al.* [48] claimed that as the depth of a neural network increases, the gradients resemble white noise, leading to meaningless weight updates. In the same study, the authors found that the residual network (ResNet) proposed in [2] could prevent the shattering of gradients.

4.2.2 Residual Learning

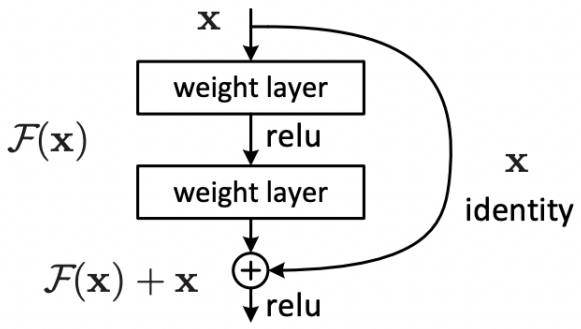

Figure 4.4: Structure of the residual block in ResNet [2].

To tackle the degradation problem, He *et al.* [2] proposed an neural network architecture named ResNet, which stands for residual network. ResNet assumes that asymptotically approaching a target function $H(x)$ is equivalent to approaching the residual function $F(x) = H(x) - x$. Thus, the target function becomes $F(x) + x$. Based on this, a structure named residual block was implemented using shortcut connections (Figure 4.4). In this study ResNet18 was used as benchmark. ResNet18 consists of 18 layers: 1 2D convolutional layer with 64 kernels of size $7x7$, followed by 8 residual blocks, where the number of kernels doubles after every 2 blocks. The final output layer is a fully-connected layer with N neurons, where N corresponds the number of classes. Figure 4.5 shows the architecture of ResNet18 adapted for this study. An adaptive average pooling layer was

added at the beginning to scale the dimension of input data to 224×224, which is required input size of canonical ResNet18. In the ResNet architecture, the shortcut connections linking two residual blocks with different input/output dimensions (dashed lines in Figure 4.5) perform up-sampling operation by convolution with a unit size kernel to avoid shape mismatch.

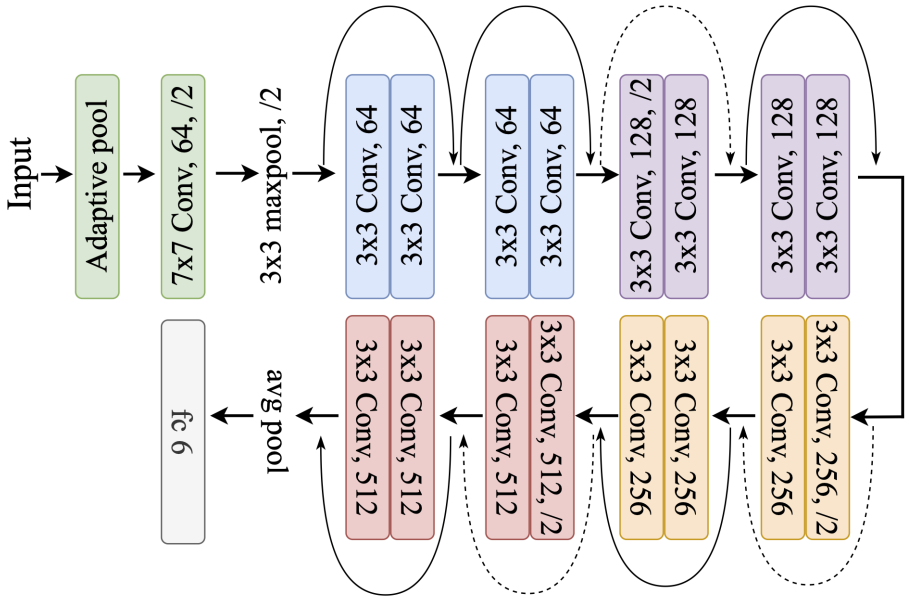

Figure 4.5: ResNet18 architecture with the addition of an adaptive average pooling layer, which scales any input to the size of $(224, 224)$, removing the input dimension limitation.

4.2.3 Knowledge Distillation

The great performance of deep neural networks comes at the cost of high computational complexity. Vanilla Knowledge distillation (KD) proposed by Hinton *et al.* [49], an offline model compression technique, was used in this study to compress and transfer the knowledge learned by ResNet18 to a shallower network. The vanilla KD involves two stages: 1) train an optimal network, which acts as teacher, with unlimited resource provision; 2) train the a simpler student network which attempts to match the predictions from teacher network during training. Conventionally, deep learning methods use cross-entropy H as loss function for classification tasks. In vanilla KD, H and the KL divergence loss D_{KL} between the class probability distribution predicted by student and teacher are weighted

averaged (Equation 4.8):

$$L_{KD} = \alpha t^2 D_{KL}(\sigma(\mathbf{z_s}, T = t), \sigma(\mathbf{z_t}, T = t)) + (1 - \alpha) H(y, \sigma(\mathbf{z_s}, T = 1)) \tag{4.8}$$

where T is the temperature used in the SoftMax function, which converts the vector of logits $\mathbf{z}$ output by the final layer into a vector of probabilities $\mathbf{p}$, where the probability of each class is calculated using:

$$p_i = \frac{e^{z_i/T}}{\sum_j e^{z_j/T}} \tag{4.9}$$

4.2.4 Proposed Method: KD-sCNN

In this study, a shallow CNN (sCNN) consisting of few CNN layers was proposed as student network. The architecture of proposed method, referred to as KD-sCNN, is shown in Figure 4.6. Note that only the structure within the yellow block is retained after training stage. The sCNN architecture consisted of 3 stacked CNN layers (3×3 kernels) with interleaving 2×8 max-pooling layers. The size was pooling layer was chosen due to the rectangular shape of input data.

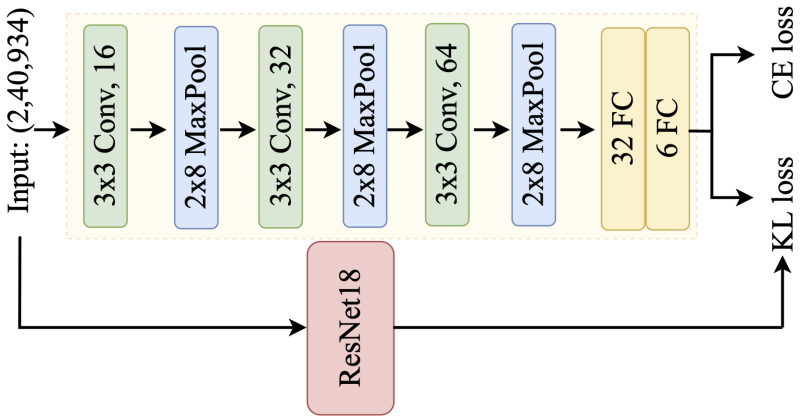

Figure 4.6: The proposed KD-sCNN architecture is shown in the yellow block.

4.3 Training Strategies

For SVM and RF, the dataset was split into a training set and a test subset in a ratio of 9 : 1, while for all aforementioned deep learning methods, the dataset was firstly split into a training and a test subset with a ratio of 9 : 1, and then 20% of training subset was reserved as validation subset. Theoretically, the samples should have the same distribution, although being collected from different subjects. However, this is usually not the case in the real-world due to domain shift [50]. As such, two training strategies were used and evaluated: standard and leave-one-subject-out strategy. The protocol of the standard strategy was:

- shuffle the dataset such that samples from various subjects are mixed.
- split the dataset into test and training subsets in a 1 : 9 ratio.
- train the classifier; for conventional classifier, 5-fold cross-validation [51] is used for tuning model parameters, while deep learning classifier uses a validation subset separated from the training set in a 2 : 8 ratio.
- evaluate the classifier on the test subset.

The protocol of leave-one-subject-out strategy differed in terms of data partitioning:

- reserve data samples from Subject i as test subset, and the rest of samples serves as training subset, which also results in a ratio of 1 : 9.
- shuffle the training subset.
- train the classifier using the same parameter-tuning method in standard strategy.
- evaluate the classifier on the test subset.

By comparing the performance of classifiers trained using these two strategies, it was possible to determine whether samples from a particular subject was domain-shifted, and assess the robustness of methods toward such shift. Lastly, to assess the methods in

terms of robustness to radar displacement, the leave-one-channel-out training strategy was employed:

- shuffle the entire dataset
- split the 2-channel dataset into single-channel training and testing subset
- train the classifier using the same parameter-tuning method in standard strategy.
- evaluate the classifier on the test subset.

The performance of classifier on the test subset could reveal the effect a positional change of radar, and how each classifier would behave under such effect. Note that the setting involved during the training of deep learning methods were kept constant across all methods: Adam optimiser with initial learning rate of 0.001 was used; batch size and training epochs were 32 and 20 respectively; the temperature used for KD was 10.

4.4 Method Evaluation

Algorithm 1 calculation of cross-test accuracy

Require: $n \geq 2$
Require: $D = \{D_1, ..., D_n\}$
Ensure: Average accuracy across test subjects $A_{average}$
  $N \leftarrow n$
  $A_{total} \leftarrow 0$
  for $i \leftarrow 1$ to N **do**
    for $j \leftarrow 1$ to N **do**
      $D_{test} \leftarrow D[j]$ ▷ Reserve Subject j as test subset
      if $i \neq j$ **then**
        $D_{train} \leftarrow \text{Stack}(D_{train}, D[i])$ ▷ Combine data from others
      end if
    end for
    Classifier $\leftarrow$ Classifier.Train(D_{train})
    $A_i \leftarrow$ Classifier.Estimate(D_{test})
    $A_{total} \leftarrow A_{total} + A_i$
  end for
  $A_{average} \leftarrow \frac{A_{total}}{N}$ ▷ This is the cross-test accuracy

For all classifiers trained on the standard strategy, 4 metrics, namely validation accuracy, test accuracy, model size and average inference time per sample, were used for

evaluate. The metrics were defined differently for conventional and deep learning methods, and they were summarised in Table 4.1, where the average inference time per sample was measured on a Intel Xeon CPU @ 2.30 GHz provided by Google Colab. Conventional classifiers trained on leave-one-subject-out strategy was evaluated using a cross-test accuracy, which follows the idea of k-fold cross-validation method [51] (Algorithm 1). Two subjects that resulted in the lowest test accuracy in leave-one-subject-out strategy were used to evaluate the deep learning models.

Table 4.1: Definition of performance evaluation metrics

Metric	Conventional Methods	Deep Learning Methods
Validation Accuracy	5-fold cross-validation accuracy	Accuracy on validation set
Test Accuracy	Accuracy on test set	Accuracy on test set
Model Size	Size after saving as `.pickle` file	Size after saving as `.pth` file
Inference Time	Mean time for one inference	Mean time for one inference

Chapter 5

Result and Discussion

THIS chapter will compare the performance of conventional classifiers versus deep learning method trained using the 3 proposed strategies, and evaluate them according metrics explained in Chapter 4.

5.1 Visualisation of Extracted Features

Channel-wise 2D-2D PCA was applied to the data samples, and the Scree plots were generated. For both channels, the "elbow" point were found at 10^{th} and 12^{th} PCs along the column and row Scree plots, respectively. For both channels, 10 column PCs explained nearly 30% of total variance (Figure 5.1(a,c)), while the 12 row PCs explained approximately 83% of variance (Figure 5.1(b,d)). Note that for the column Scree plot, only the first 100 PCs were shown, to ease the identification of "elbow". Although the "elbow" of Scree plot was found, due to the redundant information residing in the rest of 924 PCs, the portion of variance explained up to the elbow PCs was low. However, this should not cause any harm to the useful information that mainly resided before the "elbow". By performing 2D-2D PCA with $d = 12$ and $q = 10$, the each channel of the each original samples $\mathbf{X}_{\mathbf{40\times 934}}$ was reduced to feature matrix $\mathbf{C}_{12\times 10}$, which was then flattened to produce a feature vector $\mathbf{c}_{120}$. By concatenating feature vectors from the 2 channels, the original sample was reduced to a feature vector $\mathbf{x}_{240}$. The down-sampled data was further

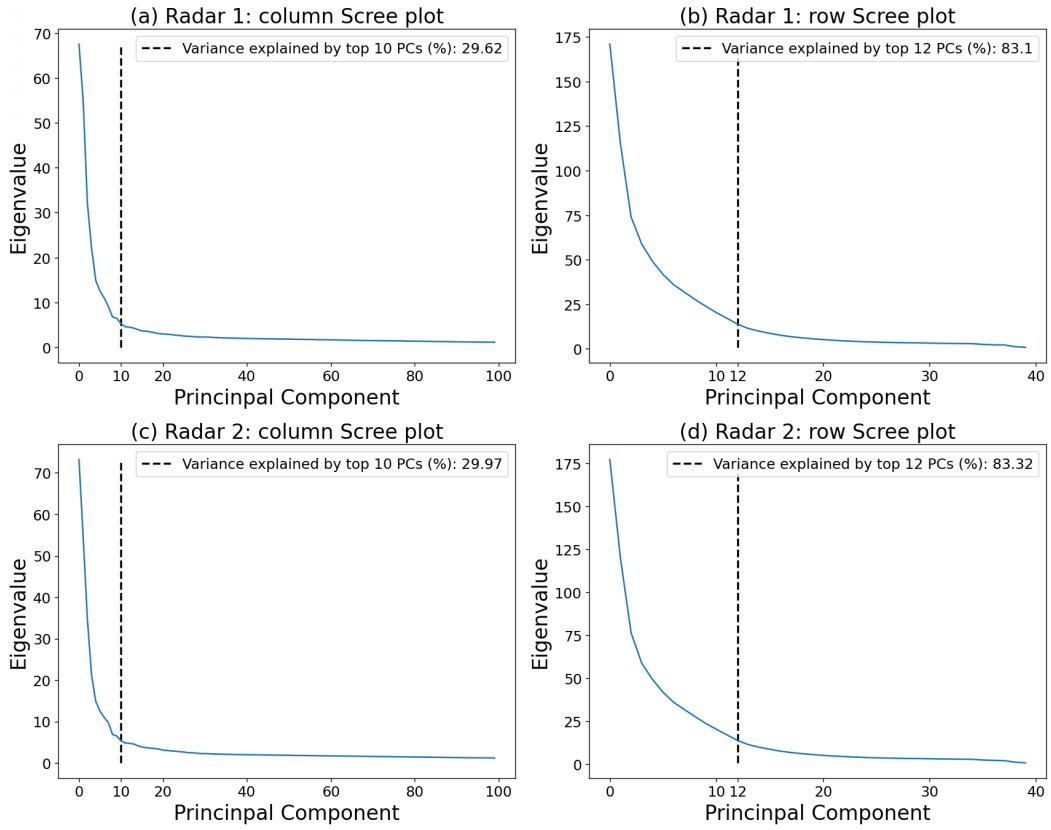

Figure 5.1: Column and row Scree plots generated by 2D-2D PCA. "Elbow" points clearly identifiable at 10^{th} column PC and 12^{th} row PC.

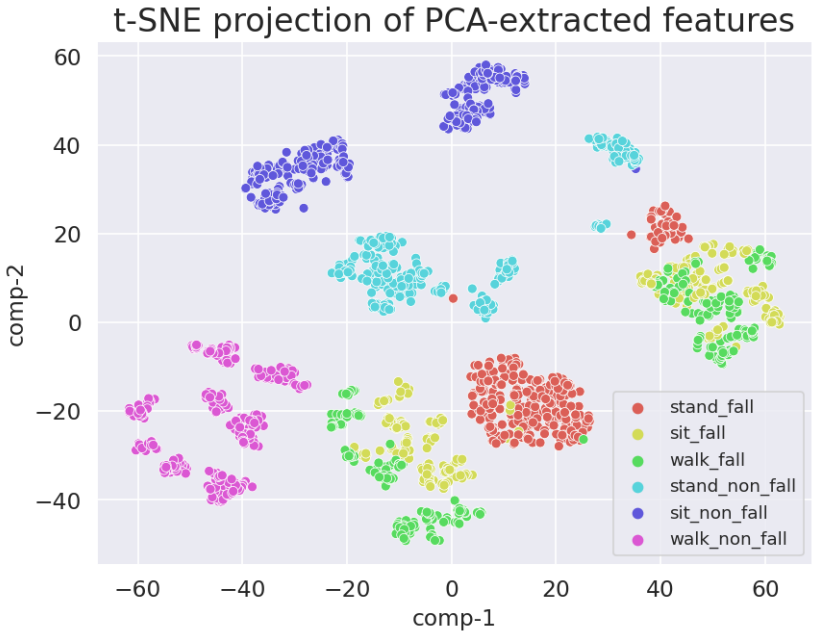

Figure 5.2: t-SNE visualisation of 240-D data on a 2-D space.

Table 5.1: Performance of SVM and RF trained using standard strategy for 6 sub-classes classification

Method	Validation Accuracy (%)	Test Accuracy (%)	Size (MB)	Inference time (ms)
SVM	**99.1**	**98.9**	**1.25**	**0.122**
RF	96.7	97.7	3.25	0.285

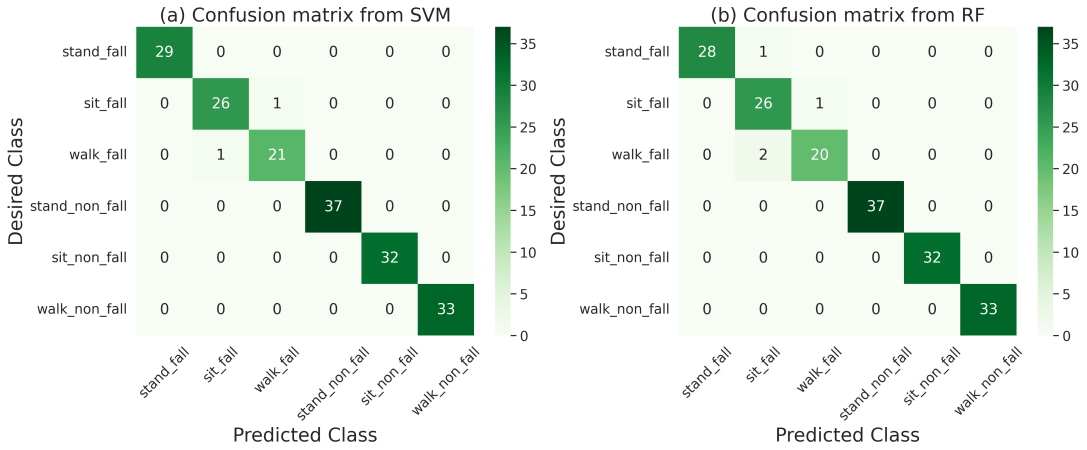

Figure 5.3: Confusion matrix generated from the predictions over test set given by SVM and RF trained by standard strategy.

reduced to 2-dimensional space using t-SNE (Figure 5.2), in which most classes appeared separable, except the *Sit Fall* and *Walk Fall*, which were mixed with each other.

5.2 Results from Classifiers Trained on Standard Strategy

The performance comparison the two classifiers was given in Table 5.1. SVM scored 98.9% and RF scored 97.7% multi-class classification accuracy in the test set. The memory occupied by SVM and RF was 1.25 MB and 3.25 MB respectively. RF resulted in larger memory footprint because it was an ensemble of 90 decision trees (optimal number found via grid search). The average inference time per sample was 0.122 ms for SVM and 0.285 ms for RF. SVM was 2.3× faster since it only require less computations. The predictions given by SVM and RF were depicted in the confusion matrices shown in (Figure 5.3), and the the binary classification accuracy was 100% for both methods, since there was no misclassification between *Fall* and *Non-Fall* classes.

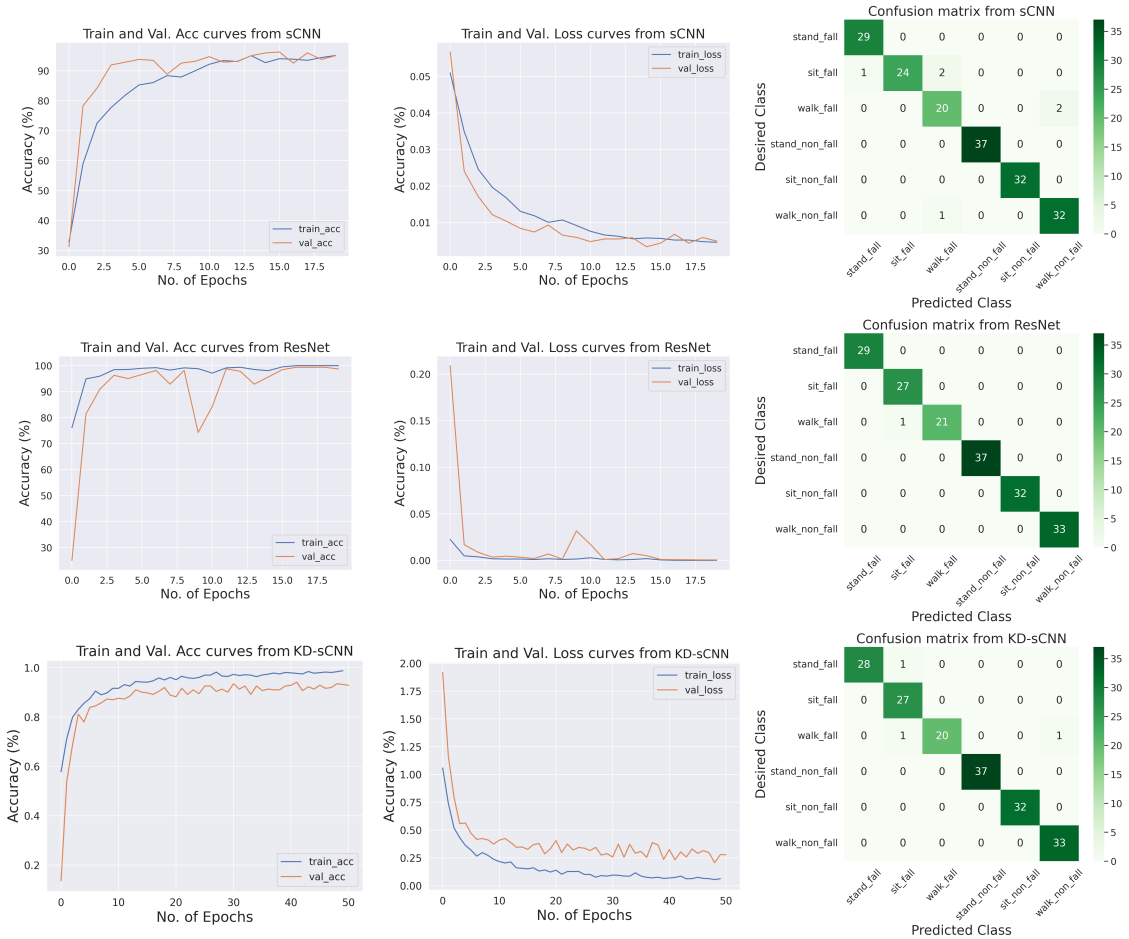

Figure 5.4: Training and validation accuracy and loss curves, and confusion matrix from sCNN (top row), ResNet18 (middle row) and KD-sCNN (bottom row) over test set.

Table 5.2: Performance of sCNN, ResNet18 and KD-sCNN trained using standard strategy for 6 sub-classes classification

Method	Test Accuracy (%)	Size (MB)	CPU inference time (ms)	GPU inference time (ms)
sCNN	96.6	0.141	4.63	1.19
ResNet18	**99.4**	43.7	74.4	3.49
KD-sCNN	98.3	**0.141**	**4.63**	**1.19**

For deep learning methods, the baseline model sCNN trained using standard strategy achieved a test accuracy of 96.6%, which was lower than the conventional classifiers. In contrast, the benchmark ResNet18 only misclassified 1 sample in the test set, resulting in the highest accuracy (99.4%) among all models, at the cost of high computational com-

plexity. The memory footprint of ResNet18 was 43.7 MB, compared to the 0.141 MB of sCNN. The average inference time of ResNet18 was around 74.4 ms on a CPU, which is more than 16× longer than sCNN. Even with the GPU acceleration, ResNet18 takes nearly 3× longer time for a inference (Table 5.2). KD-sCNN exhibited a performance between these two, achieving a test accuracy of 98.3%. There was no sign of over-fitting from the learning curves in Figure 5.4, as all 3 methods showed converging training and validation losses, with the exception that there were discrepancies between the training and validation accuracy/loss curves in KD-sCNN. For binary classification scenarios, sCNN and KD-sCNN gave 1 and 2 misclassifications, resulting in 99.4% and 98.9% accuracy respectively, while ResNet18 correctly classified all samples, as shown in the confusion matrices (Figure 5.4(right column)).

5.3 Results from Classifiers Trained on Leave-One-Subject-Out Strategy

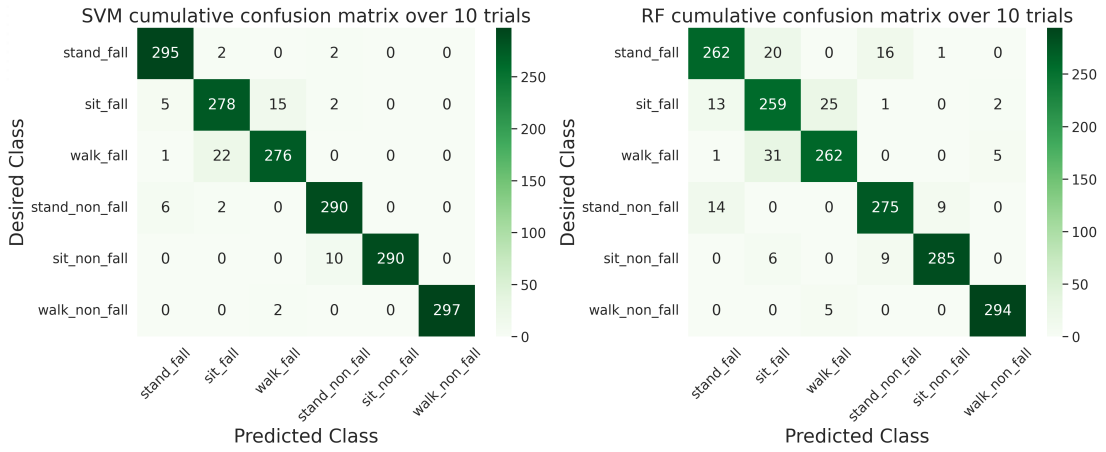

Figure 5.5: Cumulative confusion matrix over 10 trials from SVM (left) and RF (right) trained on leave-one-subject-out strategy.

The cross-subject test accuracy and average test accuracy of SVM and RF trained using leave-one-subject-out strategy are summarised in Table 5.3. The average test accuracy were 96.2% for SVM and 91.2% for RF, which were reduced by 2.7% and 6.5% compared

Table 5.3: Performance of SVM and RF trained using leave-one-subject-out strategy for 6 sub-classes classification

Trial #	SVM Test Accuracy (%)	RF Test Accuracy(%)
0	98.9	93.9
1	98.3	97.2
2	96.7	93.3
3	97.2	87.6
4	**88.3**	87.2
5	93.3	**79.9**
6	97.2	92.8
7	96.1	92.2
8	96.1	88.3
9	99.4	99.4
Average	96.2	91.2

Table 5.4: Test accuracy of sCNN, ResNet18 and KD-sCNN trained using leave-one-subject-out strategy. Data from Subject 4 and 5 were reserved as test set respectively.

Method	Test Acc. on Subject 4 (%)	Test Acc. on Subject 5 (%)
sCNN	85.5	82.1
ResNet18	93.3	96.7
KD-sCNN	90.5	87.2

to results obtained with standard strategy. According to the cumulative confusion matrices (Figure 5.5), SVM and RF score 99.2% and 97.2% in binary classification scenario. It could be also seen that SVM and RF performed differently across the 10 trials. In particular, both SVM scored lowest test accuracy (88.3%) in trial 4, and RF in trial 5 (79.9%), implying that data from Subject 4 and 5 have potentially experienced domain shift. The test accuracy of deep learning methods on the data from these two subjects are given in Table 5.4. sCNN, ResNet18 and KD-sCNN scored 85.5%, 93.3% and 90.5% on test data from Subject 4, and 82.1%, 96.7% and 87.2% on data from Subject 5, respectively. sCNN performed worse than SVM and RF on Subject 4, and with the aid of powerful ResNet18, the accuracy of KD-sCNN increased by 5.0% compared to sCNN, outperforming SVM and RF. A similar trend in accuracy was observed on test data from Subject 5, where the performance of KD-sCNN was significantly boosted by the teacher network ResNet18.

5.4 Results from Classifiers Trained on Leave-One-Channel-Out Strategy

The multi-class classification accuracy of classifiers trained on leave-one-channel-out are given in Table 5.5. Classifiers scored lowest accuracy in this scenario, and generally performed better when tested on Radar 1 data compared to Radar 2 data, except KD-sCNN. RF was the most impacted classifier, as it average test accuracy was 78.2%, which was 19.5% less than its performance on a standard training strategy, while the accuracy of ResNet18 only dropped by 8.0%. The trend of reduction aligned with the general accuracy trend observed in previous evaluation: SVM was less accurate than ResNet18 and more powerful than RF and sCNN; with the knowledge distilled from ResNet18, KD-sCNN was comparable to SVM in terms of accuracy. With regard to binary classification accuracy, RF still scored the lowest accuracy of 91.7%, while the performance of other methods was around 95%, with ResNet yielding the best result (97.2%) (Table 5.6).

Table 5.5: Performance of all classifiers trained using leave-one-channel-out strategy for 6 sub-classes classification

Method	Test Accuracy on Radar 1 (%)	Test Accuracy on Radar 2 (%)	Average Accuracy(%)
SVM	87.9	86.4	87.1
RF	80.4	76.0	78.2
sCNN	84.3	85.7	85.0
ResNet18	93.7	89.2	91.4
KD-sCNN	86.5	87.1	86.8

Table 5.6: Performance of all classifiers trained using leave-one-channel-out strategy for binary classification

Method	Test Accuracy on Radar 1 (%)	Test Accuracy on Radar 2 (%)	Average Accuracy(%)
SVM	96.1	95.6	95.8
RF	92.4	91.0	91.7
sCNN	95.1	95.2	95.2
ResNet18	98.4	95.9	97.2
KD-sCNN	95.0	96.0	95.5

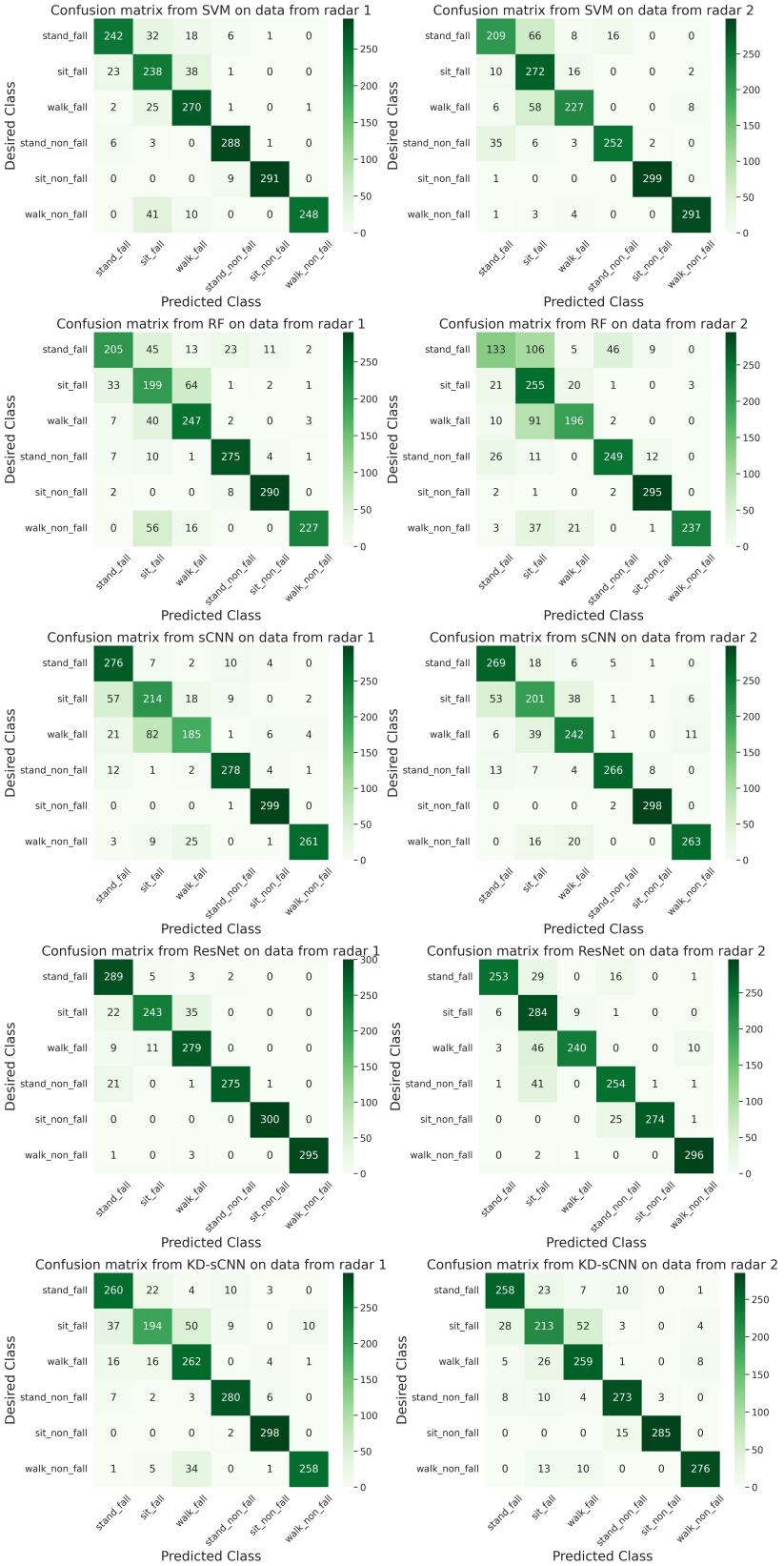

Figure 5.6: Confusion matrices of all methods trained on leave-one-channel-out strategy.

5.5 Discussion

5.5.1 General Performance of Evaluated Methods

Impact of Training Strategy to Classification Accuracy

All classifiers achieved over 96% in the multi-class classification task when the distribution of of test samples were accessible through the training samples from the same subjects. Knowing the train-test split ratio ($9:1$) and the number of training samples per subject (179 ± 1), the number of samples required for the classifiers to achieve similar performance would be ~ 150. Despite only a half (~ 75) has to be *Fall* samples, it is still impractical for target users to perform such high number of falls. As such, the results of classifiers trained on leave-one-subject-out strategy would be a more appropriate measure of system feasibility. The average accuracy of SVM and RF were decreased, however, the latter suffered a greater reduction. The impact of data from an unseen subject as test set was also significant in deep learning methods. On the least distinguishable data from 2 subjects, sCNN suffered great accuracy reduction compared to SVM and RF. However, as the effect of KD emerges, the accuracy of KD-sCNN significantly improved. Such performance boost from KD was observed in all evaluation, underling the usefulness of the technique. ResNet18 and KD-sCNN outperformed the counterparts, which is a sign of stronger ability to learn the target function, and not the specific features of each subject. Lastly, large accuracy reduction was seen in all classifier trained on leave-one-channel-out, and potential reasons include: 1) the inter-channel dependencies served as important features for classifiers trained on data from both channels; 2) the input shape was halved, leading to reduction of trainable input information, which is critical to deep learning methods; 3) the information residing in each channel differed largely due to external factors (e.g., subjects performed activities closer to one radar than the other).

Observations from Misclassifications

Looking at the distribution of misclassfications in the confusion matrices, a few observations could be made. In standard training, the very few erroneous were prediction mainly

caused by the confusion between *Sit Fall* and *Walk Fall* (Figure 5.3 and 5.4). This trend aligned with the distribution in the cumulative matrices generated from leave-one-subject-out strategy evaluation (Figure 5.5), where the misclassification mainly concentrated in these 2 sub-classes. From this observation, one could speculate that the classifiers learned the pattern of a pure fall (*Stand Fall*), but they were less capable in distinguishing the patterns of the activities before a fall (walking for *Walk Fall* and sitting down for *Sit Fall*). In leave-one-channel-out training, the misclassification concentrated in the 3 sub-classes of *Fall* class, which implies an undermined ability to discriminate the patterns in actual falls. However, the number erroneous predictions was relatively low for all model if the binary classification scenario was considered. Another interesting observation was the reduced ability to distinguish *Walk Non-Fall* from *Sit Fall* and *Sit Fall* generally observed in classifiers trained on leave-one-channel-out strategy. A potential explaination is that the activities of latter sub-classes involved some movements towards the radar, which made them appeared similar to *Walk Non-Fall* that was pure movement toward radars.

5.5.2 Feasibility of Evaluated Methods for Practical Use

Obtaining high accuracy on the development environment is fairly easy. However, the practical usefulness of a method would be a very important factor to consider. For instance, the superiority of ResNet18 under all evaluation scenario cannot compensate its cost of computation. Considering the scenario when the classifiers have to be deployed on edge devices to maximise user privacy, the computational cost becomes critical. The time of inference per sample was around 74 ms for ResNet on a 2.4 GHz CPU, which will be more than 177 ms when running at full speed on a typical commercially-off-the-shelf single board computer with 1.0 GHz CPU (e.g., the Rasberry Pi Zero W used in the radar nodes). On the other hand, SVM, RF and KD-sCNN, require much less resources. KD-sCNN occupies very small memory compared to RF and SVM. In particular, the size of SVM will increase exponentially as the training set increments. However, to run a prediction, SVM and RF require far less computations, which are inherently cross-platform (e.g., SVM only performs projections and inner products), while deep learning methods rely on the

framework provider to choose which platform is supported. For this study, the SVM and KD-sCNN classifier have been found attractive in practice due to their high performance and low computational overhead. The overall system have outperformed other solutions from existing systems in terms of test accuracy in standard training strategy. Promising results were also obtained by the proposed system in dealing with domain shift and sensor displacement, demonstrating high robustness.

Chapter 6

Conclusion and Future Works

6.1 Conclusion

In this study, methods were proposed to tackle the challenges of indoor fall detection based on a multistatic UWB radars system. A dataset consisting of 1795 samples was generated from 10 subject and pre-processed using high-pass filtering that effectively improved SNR. Both conventional machine learning classifier, SVM and RF, and deep learning classifiers, sCNN, ResNet18 and KD-sCNN, were trained and evaluated under different settings. For conventional classifiers, instead of relying on manual features, the unsupervised feature extraction method PCA was performed to automatically select the most useful features, which further enhance the generalisation ability of classifiers. On the other hand, deep learning method were trained directly on raw data due to their self-contained automatic feature extraction ability. Three training strategies, namely standard, leave-one-subject-out and leave-one-channel-out strategy, were employed, which yielded differential results for each classifier, underlining their respective ability to deal with the constraint given in each strategy. A comprehensive analysis and discussion was made on the results, from which a few interesting observation were made. SVM and ResNet18 were the most robust classifiers as smaller accuracy reduction was observed. After considering the requirement for practical uses, this study concluded that the conventional classifier SVM and the knowledge-distilled shallow neural network KD-sCNN, which achieved respectively scored

98.9% and 98.3% test accuracy when trained under standard strategy. The results of the proposed system were comparable to the best known system, while keeping the computational cost to a minimum by sampling at only 10 Hz.

6.2 Limitation and Future Works

The major limitation of this study was the number and age distribution of recruited participants. Since only 10 participants were recruited, the statistical significance of this study is not absolutely guaranteed. Also, the participants were all young individuals with potentially different physical and health condition compared to the target population, which might further undermine the validity of proposed system on the target users. Another limitation was the room size, as it was not possible to evaluate the proposed system could detect falls happened distantly. For future works, it is recommended to repeat the experiments after addressing the mentioned limitations. In addition, since the radar have penetration ability, it will be interesting to evaluate the system performance with occlusion between radar and participant.

Bibliography

[1] G. Y. Kanyongo, "Determining the correct number of components to extract from a principal components analysis: a monte carlo study of the accuracy of the scree plot," *Journal of Modern Applied Statistical Methods*, vol. 4, no. 1, p. 13, 2005.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[3] B. Moreland, R. Kakara, and A. Henry, "Trends in nonfatal falls and fall-related injuries among adults aged 65 years—united states, 2012–2018," *Morbidity and Mortality Weekly Report*, vol. 69, no. 27, p. 875, 2020.

[4] E. Burns and R. Kakara, "Deaths from falls among persons aged 65 years—united states, 2007–2016," *Morbidity and Mortality Weekly Report*, vol. 67, no. 18, p. 509, 2018.

[5] C. G. Moran, R. T. Wenn, M. Sikand, and A. M. Taylor, "Early mortality after hip fracture: is delay before surgery important?" *JBJS*, vol. 87, no. 3, pp. 483–489, 2005.

[6] K. De Miguel, A. Brunete, M. Hernando, and E. Gambao, "Home camera-based fall detection system for the elderly," *Sensors*, vol. 17, no. 12, p. 2864, 2017.

[7] Z.-P. Bian, J. Hou, L.-P. Chau, and N. Magnenat-Thalmann, "Fall detection based on body part tracking using a depth camera," *IEEE journal of biomedical and health informatics*, vol. 19, no. 2, pp. 430–439, 2014.

[8] S. B. Khojasteh, J. R. Villar, C. Chira, V. M. González, and E. De la Cal, "Improving fall detection using an on-wrist wearable accelerometer," *Sensors*, vol. 18, no. 5, p. 1350, 2018.

[9] Q. Wu, Y. D. Zhang, W. Tao, and M. G. Amin, "Radar-based fall detection based on doppler time–frequency signatures for assisted living," *IET Radar, Sonar & Navigation*, vol. 9, no. 2, pp. 164–172, 2015.

[10] T. Xu, Y. Zhou, and J. Zhu, "New advances and challenges of fall detection systems: A survey," *Applied Sciences*, vol. 8, no. 3, p. 418, 2018.

[11] A. Bannon, A. Rapeaux, and T. G. Constandinou, "Tiresias: A low-cost networked uwb radar system for in-home monitoring of dementia patients," in *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. IEEE, 2021, pp. 7068–7072.

[12] N. Andersen, K. Granhaug, J. A. Michaelsen, S. Bagga, H. A. Hjortland, M. R. Knutsen, T. S. Lande, and D. T. Wisland, "A 118-mw pulse-based radar soc in 55-nm cmos for non-contact human vital signs detection," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 12, pp. 3421–3433, 2017.

[13] J. Li, Z. Zeng, J. Sun, and F. Liu, "Through-wall detection of human being's movement by uwb radar," *IEEE Geoscience and Remote Sensing Letters*, vol. 9, no. 6, pp. 1079–1083, 2012.

[14] O. D. Lara and M. A. Labrador, "A survey on human activity recognition using wearable sensors," *IEEE communications surveys & tutorials*, vol. 15, no. 3, pp. 1192–1209, 2012.

[15] A. K. Bourke, J. O'brien, and G. M. Lyons, "Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm," *Gait & posture*, vol. 26, no. 2, pp. 194–199, 2007.

[16] Q. Li, J. A. Stankovic, M. A. Hanson, A. T. Barth, J. Lach, and G. Zhou, "Accurate, fast fall detection using gyroscopes and accelerometer-derived posture information,"

in *2009 Sixth International Workshop on Wearable and Implantable Body Sensor Networks*. IEEE, 2009, pp. 138–143.

[17] S. Poddar, V. Kumar, and A. Kumar, "A comprehensive overview of inertial sensor calibration techniques," *Journal of Dynamic Systems, Measurement, and Control*, vol. 139, no. 1, 2017.

[18] S. Shirmohammadi and A. Ferrero, "Camera as the instrument: The rising trend of vision based measurement," *IEEE Instrumentation & Measurement Magazine*, vol. 17, no. 3, pp. 41–47, 2014.

[19] V. Lempitsky, P. Kohli, C. Rother, and T. Sharp, "Image segmentation with a bounding box prior," in *2009 IEEE 12th international conference on computer vision*. IEEE, 2009, pp. 277–284.

[20] Z. Gang and Q. Long, "Silhouette extraction from multiple images of an unknown background," in *Proceedings of the Asian Conference of Computer Vision, Citeseer*. Citeseer, 2004.

[21] M. Saeedkondori, *Estimation of the center of gravity of the human body using image processing*.

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

[23] G. Diraco, A. Leone, and P. Siciliano, "An active vision system for fall detection and posture recognition in elderly healthcare," in *2010 Design, Automation & Test in Europe Conference & Exhibition (DATE 2010)*. IEEE, 2010, pp. 1536–1541.

[24] Z. Zhang, C. Conly, and V. Athitsos, "Evaluating depth-based computer vision methods for fall detection under occlusions," in *International symposium on visual computing*. Springer, 2014, pp. 196–207.

[25] M. A. Richards, J. Scheer, W. A. Holm, and W. L. Melvin, *Principles of modern radar*. Citeseer, 2010, vol. 1.

[26] G. R. Aiello and G. D. Rogerson, "Ultra-wideband wireless systems," *IEEE microwave magazine*, vol. 4, no. 2, pp. 36–47, 2003.

[27] D. Wang, S. Yoo, and S. H. Cho, "Experimental comparison of ir-uwb radar and fmcw radar for vital signs," *Sensors*, vol. 20, no. 22, p. 6695, 2020.

[28] H. Yoshino, V. G. Moshnyaga, and K. Hashimoto, "Fall detection on a single doppler radar sensor by using convolutional neural networks," in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*. IEEE, 2019, pp. 2889–2892.

[29] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: Lstm cells and network architectures," *Neural computation*, vol. 31, no. 7, pp. 1235–1270, 2019.

[30] A. Mol, P. T. S. B. Hoang, S. Sharmin, E. M. Reijnierse, R. J. van Wezel, C. G. Meskers, and A. B. Maier, "Orthostatic hypotension and falls in older adults: a systematic review and meta-analysis," *Journal of the American Medical Directors Association*, vol. 20, no. 5, pp. 589–597, 2019.

[31] A. Blake, K. Morgan, M. Bendall, H. Dallosso, S. Ebrahim, T. a. Arie, P. Fentem, and E. Bassey, "Falls by elderly people at home: prevalence and associated factors," *Age and ageing*, vol. 17, no. 6, pp. 365–372, 1988.

[32] N. V. Chawla, N. Japkowicz, and A. Kotcz, "Special issue on learning from imbalanced data sets," *ACM SIGKDD explorations newsletter*, vol. 6, no. 1, pp. 1–6, 2004.

[33] T. Han, W. Kang, and G. Choi, "Ir-uwb sensor based fall detection method using cnn algorithm," *Sensors*, vol. 20, no. 20, p. 5948, 2020.

[34] P. Sharma, B. Kumar, D. Singh, and S. Gaba, "Critical analysis of background subtraction techniques on real gpr data." *Defence Science Journal*, vol. 67, no. 5, 2017.

[35] Y. Wang, J. Zhou, J. Tong, and X. Wu, "Uwb-radar-based synchronous motion recognition using time-varying range–doppler images," *IET Radar, Sonar & Navigation*, vol. 13, no. 12, pp. 2131–2139, 2019.

[36] M. Sheykhmousa, M. Mahdianpari, H. Ghanbari, F. Mohammadimanesh, P. Ghamisi, and S. Homayouni, "Support vector machine versus random forest for remote sensing image classification: A meta-analysis and systematic review," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 6308–6325, 2020.

[37] S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie, "High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning," *Pattern Recognition*, vol. 58, pp. 121–134, 2016.

[38] D. R. Cutler, T. C. Edwards Jr, K. H. Beard, A. Cutler, K. T. Hess, J. Gibson, and J. J. Lawler, "Random forests for classification in ecology," *Ecology*, vol. 88, no. 11, pp. 2783–2792, 2007.

[39] D. Zhang and Z.-H. Zhou, "(2d) 2pca: Two-directional two-dimensional pca for efficient face representation and recognition," *Neurocomputing*, vol. 69, no. 1-3, pp. 224–231, 2005.

[40] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[41] J. Pareek and J. Jacob, "Data compression and visualization using pca and t-sne," in *Advances in Information Communication Technology and Computing*. Springer, 2021, pp. 327–337.

[42] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, and A. Lopez, "A comprehensive survey on support vector machine classification: Applications, challenges and trends," *Neurocomputing*, vol. 408, pp. 189–215, 2020.

[43] A. Widodo and B.-S. Yang, "Support vector machine in machine condition monitoring and fault diagnosis," *Mechanical systems and signal processing*, vol. 21, no. 6, pp. 2560–2574, 2007.

[44] P. Lameski, E. Zdravevski, R. Mingov, and A. Kulakov, "Svm parameter tuning with grid search and its impact on reduction of model over-fitting," in *Rough sets, fuzzy sets, data mining, and granular computing*. Springer, 2015, pp. 464–474.

[45] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[46] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[47] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.

[48] D. Balduzzi, M. Frean, L. Leary, J. Lewis, K. W.-D. Ma, and B. McWilliams, "The shattered gradients problem: If resnets are the answer, then what is the question?" in *International Conference on Machine Learning*. PMLR, 2017, pp. 342–350.

[49] G. Hinton, O. Vinyals, J. Dean *et al.*, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, vol. 2, no. 7, 2015.

[50] T. Tommasi, M. Lanzi, P. Russo, and B. Caputo, "Learning the roots of visual domain shift," in *European Conference on Computer Vision*. Springer, 2016, pp. 475–482.

[51] T. Fushiki, "Estimation of prediction error by using k-fold cross-validation," *Statistics and Computing*, vol. 21, no. 2, pp. 137–146, 2011.