Computer Vision Based Posture Estimation and Fall Detection

Kripesh Adhikari

This document is submitted in partial fulfilment of the requirements of Bournemouth University of Doctor of Philosophy (PhD)

Doctor of Philosophy

Bournemouth University

July, 2019

Copyright statement

Contents

List of Figures

List of Tables

Abstract

Falls are a major health problem, especially in the elderly population. Increasing fall events demands a high quality of service and dedicated medical treatment which is an economic burden. Serious injuries due to fall can cost lives in the absence of immediate care and support. Therefore, a monitoring system that can accurately detect fall events and generate instant alerts for immediate care is extremely necessary. To address this problem, this research aims to develop a computer vision-based fall detection system. This study proposes fall detection in three stages: (A) Detection of human silhouette and recognition of the pose, (B) Detection of the human as three regions for different postures including fall and (C) Recognise fall and non-fall using locations of human body regions as distinguishing features. The first stages of work comprise human silhouette detection and identification of activities in the form of different poses. Identifying a pose is important to understand a fall event where a change of pose defines its characteristics. A fall event comprises of sequential change of poses and ends up in a lying pose. Initial pose during a fall can be standing, sitting or bending but the final pose is usually a lying pose. It would, therefore, be beneficial if lying pose is recognised more accurately than other normal activities such as standing, sitting, bending or crawling to address a fall. Hence in the first stage, Background Subtraction (BS) is used to detect human silhouette. After background subtraction, the foreground images were used in a Convolutional Neural Network (CNN) to recognise different poses. The RGB and the Depth images were captured from a Kinect Sensor. The fusion of RGB and Depth images were explored for feeding to a convolutional neural network. Depth together with RGB complimented each other to overcome their weakness respectively and proved to be a significant strategy. The classification was performed using CNN to recognise different activities with 81% accuracy on validation. The other challenge in fall detection is the tracking of a person during a fall. Background Subtraction is not sufficient to track a fallen person especially when there are lighting and viewpoint variations in the environment and present of another object like furniture, a pet or even another person. Furthermore, tracking becomes tougher during the fall in comparison to normal activities like walking or sitting because the rate of change pose is higher during a fall. To overcome this, the idea is to locate the regions in the body in every frame and consider it as a stable tracking strategy. The location of the body parts provides crucial information to distinguish falls from the other normal activities as the person is detected all the time during these activities. Hence the second stage of this research consists of posture detection using the pose estimation technique. This research proposes to use CNN based pose estimation using simplified human postures. The available joints are grouped according to three regions: Head, Torso and Leg and then finally fed to the CNN model with just three inputs instead of several available joints. This strategy added stability in pose detection and proved to be more effective against complex poses observed during a fall. To train the CNN model, transfer learning technique was used. The model was able to achieve 96.7% accuracy in detecting the three regions on different human postures on the publicly available dataset. A system which considers all the lying poses as falls can also generate a higher false alarm. Lying on bed or sofa can easily generate a fall alarm if they are recognised as falls. Hence, it is important to recognise actual fall by considering a sequence of frames that defines a fall and not just the lying pose. In the third and final stage, this study proposes Long Short-Term Memory (LSTM) recurrent networks-based fall detection. The proposed LSTM model uses the detected three region's location as input features. LSTM is capable of using contextual information from the sequential input patterns. Therefore, the LSTM model was fed with location features of different postures in a sequence for training. The model was able to learn fall patterns and distinguish them from other activities with 88.33% accuracy. Furthermore, the precision of the fall class was 1.0. This is highly desirable in the case of fall detection as there is no false alarm and this means that the cost incurred in calling medical support for a false alarm can be completely avoided.

Acknowledgements

First of all, I would like to express my gratitude to my supervisors Dr Hammadi Nait-Charifand Prof. Hamid Bouchachia for their continuous support and advice.

I would like to thank Dr Rudra Poudel who is a friend and senior who always guided me throughout my research and motivated me to undertake a PhD and my friend Andrew Yearp for his support during my study.

Finally, I would like to thank my wife Lata Khanal who have supported me unconditionally and gave me this opportunity to study, my parents Krishna Adhikari and Durga Adhikari whose blessings are always with me and my son my lucky charm Kriansh Adhikari.

Declaration

This report has been created by myself and has not been submitted in any previous application for any degree. The work in this document has been undertaken by myself except where otherwise stated. The materials related to activity recognition for indoor fall detection using convolutional neural network have been published in Adhikari et al. (2017)

Chapter 1

Introduction

1.1 Research Aim

The aim of this research is to propose a computer vision based posture estimation and fall detection. Posture estimation is able to detect human and can be used as a detection based tracking strategy. It is necessary to track a fallen person under all the challenging conditions such as lighting variations, occlusions, scaling factors, multi-views and also the presence of another moving object which may be a pet, a toy or another person. Different activities are recognised in the initial stage with the aim that it can represent a fall when observed in a consecutive sequence. The detected postures have specific location information which can be then used as unique features for distinguishing fall from other normal activities like standing, sitting, lying, bending or crawling.

1.2 Research Objective

The main objective of this research is to develop an indoor camera-based detection system which can:

- Detect a human silhouette overcoming different challenges such as lighting variations, partial occlusion in the presence of furniture and the presence of other moving objects.
- Track a person in a real-life scenario during different activities.
- Recognise different poses and classify them into: Standing, Sitting, Lying, Bending and Crawling.
- Distinguish fall from other normal activities.

1.3 Motivation

1.3.1 Background

Fall can be described as an unintentional or sudden change of position of the body from an upright, sitting or lying position to a lower horizontal position Noury et al. (2007). According to Hyndman et al. (2002), a fall is an event that results in a person coming to rest on the ground or any lower level unintentionally. They are uncertain and can occur even at minimum risk. In a systematic review presented by (Heinrich et al. (2010),p.895), the other definitions of fall were:

”Fall is an unintentional loss of balance causing unexpected contact with the ground” and

”A fall is an unintended change in person’s position related to standing or sitting or lying with or without any sign of injury”.

Actually, all the definitions of fall are derived from its characteristics. During a fall, the rate of change of pose is higher than normal activities like standing, sitting or walking. Acceleration of the body is higher during fall and therefore the person can get severely injured if the fall takes place from a height and on the hard surface. Most of the fall also ends up in lying. One can assume that a fall might have occurred if a person is found simply lying on a floor. It is not normal for a person lying on the floor while there is a bed or sofa available in a room. Besides that, in many cases, a fall is confirmed when no motion is observed after a lying pose for a certain period of time. All these characteristics indicate that a fall is an abnormal activity and has distinguishing characteristics from other normal activities. Fall can lead to a different level of injuries and can be even more serious if immediate help is not available. This can lead to even death ultimately in the absence of any support. Therefore, it is very important to look into a fall detection system that can support people by whistle-blowing immediately after a fall.

1.3.2 Fall as the biggest threat to elderly

People aged above 65 can be considered as elderly or old Sylliaas et al. (2009). Elderly people have a high falling rate and this rate of fall is increasing every year Bian et al. (2015). About 30% of people over the age of 65 falls at least once and in half of the cases, falls are recurrent Kangas et al. (2009); Dionyssiotis (2012). The major reason behind fall is the 'ageing' of the population which is also increasing every year. Ageing of a population refers to the increase in the number of elderly people. According to the report presented by Dunnell (2008), over the last 25 years, the number of people aged 65 and over in the UK has increased by 16%. In addition, the number of population aged 85 and above has shown the highest growth in population.

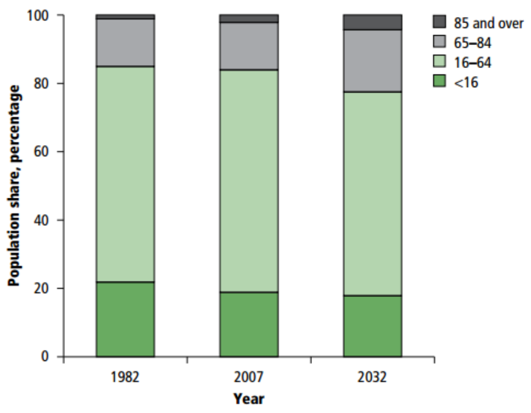


Figure 1.1: *Population age structure, 1982-2032,UK Dunnell (2008)*

.

According to the statistics shown in figure 1.1, the number of the oldest age group which is represented by the age of 85 and over have risen by nearly 680,000 to reach 1.3 million in 2007. The oldest age group represented 1.1 percent of the total population in 1982 which has doubled in number by 2007. By 2032, the number in the oldest age group is projected to increase by more than double reaching 3.1 million and representing 4% of the total population. National population projections indicate that population ageing will continue for the next few decades. Hence, ageing produces a great threat leading to falling which has been recorded very high among the oldest age CDC (2015).

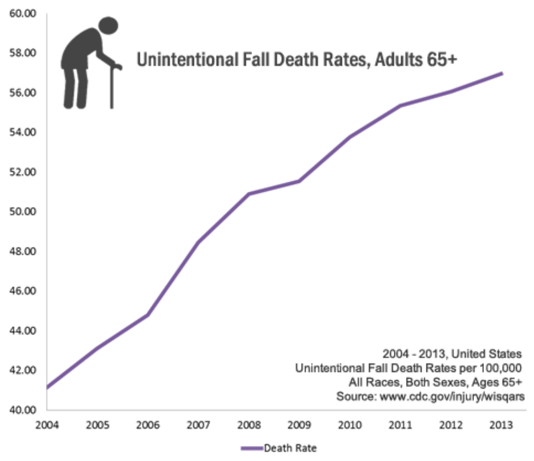


Figure 1.2: *Unintentional fall death rates per 100,000 aged above 65 CDC (2015)*

.

Here is another statistics presented in figure 1.2 on a number of unintentional fall death recorded between 2004-2013 in the USA. The deaths due to falls are exponentially rising which is a major worry. In addition to that, annually $34 billion are spent on medical costs for fall injuries in the USA. Globally, fall is one of the biggest threat to the elderly and costly in medication WHO (2007). A systematic review on the cost of falls in old age was presented by Heinrich et al. (2010) who emphasised that falls are the biggest economic burden to the society and that the fall-related costs ranged between 0.85% and 1.5% of the total healthcare expenditures in the USA, Australia and the United Kingdom. The paper also mentioned that considering the ageing societies, the total burden is likely to be increasing and therefore efforts should be directed towards developing a fall prevention programme aiming at reducing fall-related injuries. According to the fact sheet presented by Age (2016) updated February 2016, around 70,000-75,000 hip fractures occur in the UK each year mainly due to falls. The annual cost for all hip fractures in the UK including medical and social care is about £2 billion. There are higher percentages of deaths in hospital after being admitted for a fall than for all admissions mainly between the age group of 15 to 80 and above Age (2016). Sharif et al. (2018) mentioned that about 40% of traumatic injuries related patients in the hospital are admitted only due to falls.

Many fall incidents have caused serious injuries and even death in the past among elderly people as they are more prone to diseases such as dementia and epilepsy. The society will have to face two major threats due to ageing: firstly the increase of care to ageing people means higher investments on elderly care service and secondly it will lead to decrease in working population which will eventually bring a shortage in skilled caregivers for elderly people Fu et al. (2008). This indicate that the ageing society is one of the biggest challenging society especially for those who chose to live alone because they would require dedicated medical care. Therefore, fall detection is an essential monitoring system. A monitoring system that could accurately detect a fall and provide an alert instantaneously is extremely desirable. This approach could be helpful to reduce the waiting time for medical treatment and save lives.

1.4 Fall Classification

Identifying fall is very challenging as some fall events are similar to normal activities of daily life. For example, abruptly sitting down or lying on the sofa or going from standing position to lying down, have strong similarities to falls. There are possibilities of normal activities being miss-classified as fall by a fall detection system. It would be ideal if a caregiver is always present at the time of fall and could provide an instant alert. However, monitoring of such level is practically impossible and an automatic monitoring system instead is more desirable. Although fall detection may be generally considered as sub-section of general motion estimation, fall detection has distinguishable characteristics to that of general motion detection. There is a sudden change in height and width of the body after a fall incident and there is also an inactivity period on the floor. It can be noticeable by the sleeping pose or displacement of the head. Beside that inclination angle and velocity of the body before the fall can provide a significant clue for fall detection Yu (2008); Rougier et al. (2011); Vaidehi et al. (2011); Shoaib et al. (2010). Nait-Charif and McKenna (2004) brings light to the problem of distinguishing falls and daily activities by using a context-aware approach. They argued falls are different from normal activities by labelling people lying on the sofa as normal but people lying on the floor as abnormal. They demonstrated the tracking of human activities outside the inactivity zones (e.g. sofas and chairs) and suggested combining body pose and motion information can provide a significant clue for fall detection. Another attempt to distinguish falls from normal activities was studied by Wu (2000) where unique features of the velocity profile which are the magnitude change and the timing of the change were analysed during normal and abnormal activities. Different scenarios are considered to analyse fall as fall can be observed in a different orientation, transitional postures and acceleration of the body. Based on the orientation of the body, fall can also be classified into three different categories as referred to Zhang et al. (2015):

- Forward fall: In this case, a person falls in the forward direction with face impacting the floor.
- Backward fall: In this case, a person falls in the backward direction with the back of the head impacting the floor.
- Side fall: In this case, a person falls towards the left or right side in forwarding or backward direction.

Similarly, according to the transition of postures, fall can be classified as Yu (2008):

- Fall from standing: In this case, a fall occurs from standing still pose or during walking. This kind of fall may have a higher impact on the floor during the fall due to the higher position of head and torso region and hence can cause greater injury than other types.
- Fall from sitting: In this case, a fall occurs from sitting position and the impact on the floor during the fall is lower and so is the level of injury in comparison to fall from standing case.
- Fall from lying: In this case, a fall occurs in lying position from bed or sofa and the impact on the floor and the level of injury during the fall is also lower in comparison to fall from standing case.
- Fall from other transition postures: In this case, a fall occurs from bending or crawling position and the impact on the floor and the level of injury during the fall is also lower in comparison to fall from standing case.

Apart from these, based on the acceleration of the body, a fall can be a fast or of short duration and slow or comparatively of longer duration.

1.5 Approaches to Fall Detection and its Challenges

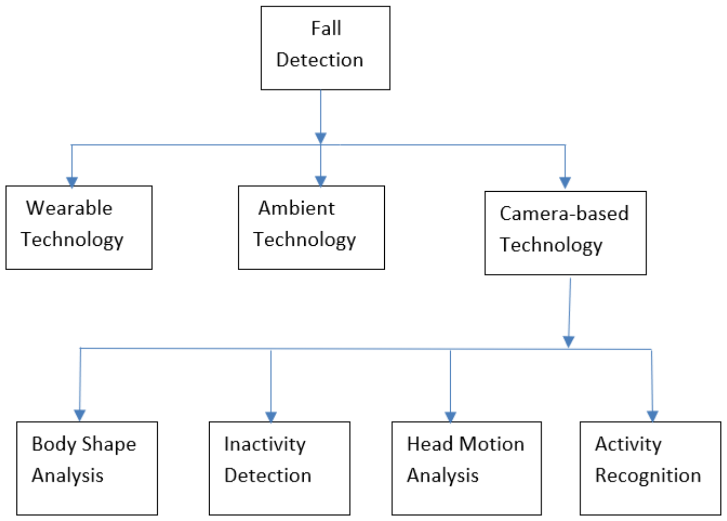


Figure 1.3: *Approaches to Fall Detection and mainly the camera based fall detection*

.

There has been a significant research carried out on developing fall detection systems in the past and most widely accepted systems according to (Noury et al. (2007); Nait-Charif and McKenna (2004); Mohamed et al. (2014); Mubashir et al. (2013)) are based on: a. Wearable technology, b. Ambient technology and c. Camera-based technology.

a. **Wearable technology-** Accelerometers, gyroscope and oscilloscopes are the examples of wearable technology used for the purpose of fall detection. A major issue with the wearable technology is discomfort as they are attached to the body. A patient who wants to move freely will have an objection to any wearable devices for monitoring a fall even if they are bulky Yu (2008). Wearable technology is highly prone to a false alarm due to similarities in action between normal activities and abnormal activities such as a case of abruptly sitting Rougier et al.

(2011). Moreover, wearing the device on the body all the time could be annoying to the patient Shoaib et al. (2010). However, the devices are more advanced these days and power or accuracy might have improved, the dilemma of being the first choice to have it on the body almost all the time still exist.

b. **Ambient technology**- Ambient technology uses installed sensors to collect data from the related person when they are near. Majority of this type of technology uses pressure sensors which sense high pressure due to the weight of the occupant at that location Yu (2008). A major drawback of such sensors is that we cannot be sure if the pressure observed is from the weight of the occupant or some other object. There are chances of false alarms due to lack of visual verification by the caregiver Jiang et al. (2013).

c. **Camera-based technology**- Camera-based technology is able to monitor activities continuously in the living environment applying image processing techniques. The video-based approach is gaining more popularity because it avoids any physical contact with the body. They are installed on building and not worn by the occupant. This method is also cost-efficient due to the recent development of inexpensive Kinect depth camera which has the capability of providing RGB image as well as depth image (Zhang et al. (2012); Gasparrini et al. (2014); Rougier et al. (2011); Bian et al. (2015)). Although there are privacy issues which have to be considered in monitoring, the use of depth can help preserve the privacy of the occupant during monitoring Zhang et al. (2012).

Comparing the three types of approaches above, we can understand that the computer vision based approach is more appropriate for developing a fall detection system. However, this approach also suffers from some shortcomings such as accuracy, occlusion, privacy and body-part similarity, variation in sizes, variation in viewpoint, variation in clothing, variation due to lighting, and variation in the background. Using appropriate image processing techniques, it is possible to overcome some of these obstacles to some extent and track the occupant activities continuously. It would be even better if we could detect the occupant, recognize the activities and distinguish a fall event from normal activities. This idea leads to the need for activity recognition and poses estimation. (Durrant-Whyte et al. (2012); Nait-Charif and McKenna (2004)) have also argued that activities such as lying on the floor is not a normal activity and can be addressed as an abnormal activity.

Recently, machine learning has set higher milestones in the field of activities recognition and human pose estimation Cao et al. (2017); Buys et al. (2014); Shotton et al. (2013); Luštrek and Kaluža (2009). Ma et al. (2014) proposed a depth-based human fall detection using machine learning approach and were able to achieve 91% sensitivity and 86% accuracy. Cao et al. (2017) is currently the state of the art in pose estimation. They have successfully performed pose estimation on real time for multiple people. They are able to predict 18 joints position in the human body. This can justify our choice of machine learning as the best approach to recognize and localise a pose. Recognition and estimation of the pose can help to distinguish a fall from other normal activities such as walking and sitting. Location of the body can be analysed in a series of frames from a video to identify different activities. It can be seen from a perspective of detection based tracking strategy of human during a fall. Therefore, the camera or vision-based fall detection approach has been further explored in the past on the basis of mainly these four techniques: Body shape analysis, Inactivity detection, Head motion analysis and Activity recognition as shown in 1.3.

- **Body shape analysis:** In this kind of approach, the human body is the prime object of interest and hence human silhouette detection and its shape analysis are mainly performed. Human silhouette or blob is acquired through image processing techniques such as foreground segmentation or background subtraction. To avoid confusion whether the foreground object in motion is human or a pet, largest area or blob is considered. A non-moving object such as furniture or wall is removed in the background. Once the human blob is achieved, the bounding box is created around the blob and height and width of the bounding box are obtained. The ratio of height and width provides an important clue that a person may be standing or lying during a fall.

- **Inactivity detection:**There can be an obvious confusion of state between sleeping and falling. Sometimes it is difficult to distinguish normal activity such as sleeping from abnormal activity such as fall. A person could fall and end up in a lying state. But then one can consider lying on a floor as a fall whereas lying on a bed or sofa as non-fall. Usually, a fall is considered when they are found lying in an uncommon region such as the floor in an active or inactive state. Many researchers have made this as an initial assumption to define a lying on a floor state as a falling state. Inactivity detection in unusual regions with the help of certain tracking algorithms have been proposed in the past for fall detection.

- **Head motion analysis:** As head being the top region of the body, it undergoes through a higher motion and distance than another part of the body, especially when analysing them during normal activity and a fall. Hence, head motion analysis can provide an important clue in fall detection. Researchers in the past have analysed head trajectory, the velocity of the head and attempted to track the motion of the head using different algorithms to distinguish between normal and abnormal activities. Floor detection is also performed to analyse the distance travelled by the head up to the floor during a fall.

- **Activity recognition:**Activity recognition has become the modern trend after setting a higher milestone and state-of-the-art with the help of Convolutional Neural Network (CNN). Different activities such as walking, jogging, sitting etc. are successfully classified with high accuracy using different machine learning algorithms. This idea can be used to distinguish normal activities from abnormal activities. Images with different pose are used to train a Neural Network (NN) and use the trained model to identify different poses of a similar class in the image. At the beginning of the development of NN, one to one image was feed to the NN. Single object and activity class-based classification were more exposed. Now with the increased computational power and different algorithms, it is possible to feed a whole sequence of images to the NN and classify a whole sequence. That means a sequence of images with poses that represents a fall can be feed to train the NN and can be used to recognise this abnormal activity. Researchers have been proposing several ways to improvise the recognition work with the help of NN.

  Similarly, pose estimation with the help of NN can help acquire location information of different body joints. Therefore, analysing the location information of the body part in different poses and actions, it is possible to use this information clues to differentiate normal activities from abnormal activities.

Moreover, to develop better understanding regarding these techniques, their capabilities and their weaknesses, a detail literature review is presented in the next chapter.

1.5.1 Fall Detection Scheme

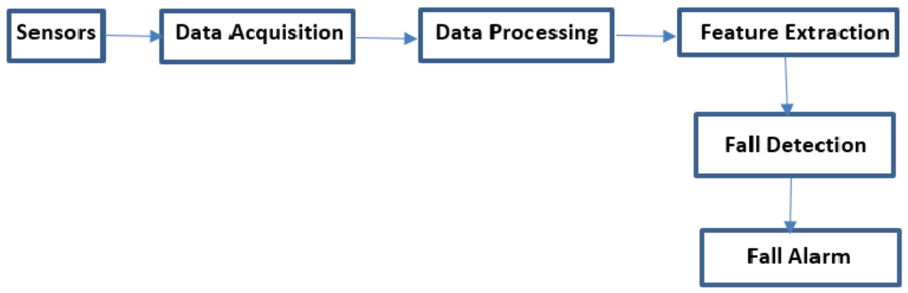


Figure 1.4: *Fall detection schematic diagram.*

A complete general scheme of a fall detection system can be illustrated as in figure 1.4. The first block represents the sensor or the type of device that is used to produce the appropriate data which will be used later by the fall detection system. Sensors may include wearable devices, accelerometers and cameras. Then the next two blocks are data acquisition and processing stages where the data is stored and prepared in such a way that they can make the most out of it when used in an algorithm. Data are stored in different sizes and formats, augmented so that crucial information can be easily extracted in the following stage of feature extraction. In the feature extraction phase, different techniques are used so that the information serves the purpose for the implementation of the fall detection algorithm. One of the major feature extraction technique is background subtraction where the object of interest is separated from the background. These features are then fed to the fall detection system which is assisted by a certain algorithm to detect falls. These algorithms may be analytical or machine learning based. Once the fall is detected, the next stage is to send an alert to the appropriate department or person that can promptly assist in this situation.

1.5.2 Challenges in Vision based Fall Detection

An initial major concern in vision-based fall is to detect human silhouette in motion. There may be different types of object in the scene. However, the aim of this study is to detect a human during a fall and thus a moving human body is the main target or object of interest. There are different challenges in computer vision to identify the object of interest from the rest of the available objects. It can be therefore be considered ideally into two important preliminary tasks:

- **Human as an object for Recognition:** A person can be considered as an object in an image. There can be different other objects in the same image. Machine learning has set a higher benchmark in image recognition and object classification. Therefore, it is possible to set the human in an image as one of the object class and attempt to classify different objects. It can hence be seen as a classification problem at the beginning of this study. But only classification or recognition of the object which is human in this study cannot provide sufficient information to analyse the occurrence of fall. Information like the height of the body, width of the body and the location of the body are some of the key information that can help to track the human body during the fall. However, recognising a pose after the detection of the human body can provide information about the occurrence of fall if the poses are analysed in each sequential frame of a video that demonstrates a fall event. If it is possible to detect the human body in every frame, it is can also be considered as detection-based tracking which can assist in visual verification of the occurrence of fall.

- **Human Silhouette Acquisition:** One of the approaches to obtain the human silhouette is background subtraction. A basic background subtraction method can help to obtain the moving object of interest in the foreground and discard the irrelevant information from the background. The objects at the background are considered static in this case. The input images considered for classification of poses in our case are also recorded in a way that a single person is the only moving object in a room. Hence background subtraction was considered. This study uses RGB and Depth images for background subtraction. They are different in terms of their composition of pixel values. An RGB image can have pixel values from 0 to 255 and Depth image are the distance values measured when the light travels from the emitter to the target. Hence background subtraction is done separately to RGB and Depth images using two different thresholds. The threshold saves the pixels values and distance values that represent the object of interest and discards all others by re-setting the values to 0 or minimum-values. The value of the threshold is considered solely on the basis of trails and error method with the attempt to get the correct values that represent the object of interest in the foreground after subtracting the background. Although the human silhouette is obtained by this approach, the output images can still have noises which may be due to change in the background, reflection, presence of shadow and lighting variations Ezatzadeh and Keyvanpour (2019). The mechanism of background subtraction is discussed in detail in the latter part of this study. The accuracy in object detection and recognition is always compromised due to the following challenges:

1. **View-point Variation:** The change in camera angle can generate different orientation of the same object in the image and thus become hard to recognise.

2. **Lighting Variation:** Lighting changes can produce a change

in the pixel values which can reduce the effect the segmentation of the object on the basis of the threshold pixel value.

3. **Scaling:** An object can appear bigger when close to the camera and smaller when it is far. Hence scaling can also affect the detection of the object.
4. **Deformation:** Deformation can also affect the detection as the object might appear in a different shape if the object of interest is not rigid.
5. **Occlusion:** There is a possibility that only a small portion of the object is visible or completely disappear. In both cases, the detection of the object is very difficult.
6. **Background Clutter:** The object of interest can sometimes blend with the environment which makes the identification of object even tougher.

A good classification model should be able to overcome all of these challenges to obtain a good accuracy in object recognition and detection purpose. Convolution Neural Network (CNN) are robust models that have set a higher benchmark in object recognition and classification. The capability and mechanism of CNN are discussed in detail in the latter part of this research.

1.6 Thesis Outline

The outlines of the remaining chapters of this report are listed below:

chapter 2 presents literature review in details. This chapter explores the work done in similar areas to understand the existing techniques and ideas proposed for fall detection. This section provides the backbone to the research. Related works are discussed in two parts: vision based fall detection and neural network based fall detection. Issues which are still a challenge in the fall detection are identified. The basic idea is to find an area where this study can contribute in the available methodology to tackle fall detection.

chapter 3 presents research aim, objective and identification of the problems in fall detection. Then a solution is proposed with a conceptual block diagram. A detail discussion of our approach is presented to provide a better understanding of the problem identified. Furthermore, this chapter presents the essential theories that form the base of our approach.

chapter 4 presents activities classification into six different classes: Standing, sitting, lying, bending, crawling and other (empty) using CNN. Furthermore, the proposed CNN architecture, dataset and experimental setups are discussed in details. This chapter also presents the discussion on theories for training and testing the model in detail. It presents a discussion on the results of the research experiment. The outputs after training, validation and test are presented and discussed in detail. The performance of the model is analysed by observing the plot in terms of accuracy and error from training and validation. The output from the test confusion matrix is also illustrated and factors such as accuracy and sensitivity are calculated. Furthermore, types of images that are correctly predicted and wrongly predicted are also displayed and the key factors influencing the result are also discussed.

chapter 5 presents pose estimation technique by transfer learning using CNN. The problem in pose detection during challenging poses are identified and a simplified human posture detection strategy is proposed to achieve a stable detection. The three different body regions are detected by grouping the available joint information representing in their respective areas. Data augmentation and training/validation settings are also discussed in details. Several publicly available datasets are used for testing under different conditions for different activities including fall. Performance is evaluated on the basis of posture detection on different activities in the publicly available dataset and compared with the output of similar work of others. Finally, some failure cases and their reasoning for the failure are also discussed.

chapter 6 presents the classification of fall from other activities based on the location features using Long Short-Term Memory (LSTM) recurrent neural network. The postures detected are stored and fed in a frame sequence of a certain length to LSTM. A certain number of consecutive frames can represent a fall, a none fall, and an empty (absence of human) scenarios. LSTM uses the contextual information from these sequences to learn a pattern to recognise a fall pattern and distinguish it from other activities include the empty sequence. The dataset used during training and the output from validation are presented. The results are discussed and compared with the existing state of the art in fall detection. Finally, a confusion matrix is presented to analyse the performance of the proposed LSTM model with precision, recall and F-score metrics.

chapter 7 presents conclusion of this research. The key findings after the analysis of the performance of the model are compared with related work done in the similar area. Furthermore, thesis contribution, limitations and future work are also presented. References are included at the end to complete the thesis.

Chapter 2

Literature Review

Several reviews exist in the literature about the fall detection system. Fall detection is not a new field of research as this area of research has been explored in the last two decades and has evolved rapidly in recent years. According to the survey presented by Xu et al. (2018), researchers have put immense effort to understand and detect fall using novel sensors and different technologies and algorithms. However, different barriers to becoming a practical application still exist. Therefore this area of research generates further research opportunities. One of the major barriers to fall detection system is the nature of the data itself. Real fall data are hardly available or inaccessible and the data that illustrates a fall event are difficult to create. There are chances of people getting real injuries while trying to create a fall event and if it is for the case of the elderly, the risk is even more. Image-based human pose classification and detection are very popular areas of research due to their wide range of applications such as video games, human-computer interaction and health care. Pose estimation can provide a very important clue to fall detection by activity recognition. Machine learning methods for pose estimation have achieved new benchmarks using complex deep convolutional network architecture Cao et al. (2017).There are mainly four approaches that were used to tackle the fall detection problem. They are mainly: Body shape analysis, Inactivity detection, Head motion analysis and Activity recognition using machine learning. The literature review is therefore discussed under these four categories.

2.1 Body Shape Analysis Based Fall:

A video based automatic fall detection method in an indoor environment using only static features of the person such as aspect ratio and inclination angle was presented by Vaidehi et al. (2011). They used background subtraction to detect human silhouette initially and then achieve static information from the human blob. Similar to the work by Vaidehi et al. (2011), a fall detection system based on human blob extraction technique using background subtraction was proposed by Gasparrini et al. (2014). They proposed a depth-based fall detection system using Kinect by tracking human blob and exploiting anthropometric relationships and features among the blobs of the human and another object. They ultimately identified fall if the human blob is near to the floor considering the relative distance between head and floor. Rougier et al. (2011) also proposed a fall detection method based on depth map video sequences. They utilised human centroid height relative to ground and body velocity considering most falls ends on the ground or near to the ground. They pointed out that velocity of the body can be a major clue for fall occurrence. Vaidehi et al. (2011) proposed an automatic fall detection method using static features that are aspect ratio and inclination angle to identify fall. However, it is arguable that the system would also identify a fall event in case a person is sleeping normally on a sofa. It is not necessary that all the fall activities end up lying horizontally on the floor. Furthermore, thresholds for height and velocity are defined as limit levels to estimate a fall activity. However, these thresholds can also be misleading even though if they were set considering a high amount of training dataset. For example, abruptly sitting on a low chair or a stumble could generate a high velocity and raise a false alarm. Therefore, it is necessary to recognise activities especially a lying pose and distinguish it from other normal activities such as standing or sitting. There are also some transitional activities such as bending and crawling that exist during fall activities. These activities also provide an important clue to analyse the direction of fall. For example, a fall event can be considered as the sequence of poses starting from standing to bending or crawling and then finally to lying. Therefore, the intention is to recognise different types of poses that give valuable information regarding after-fall posture and then analyse its static features that determine the characteristics of the fall. Ultimately with these two major steps, one can detect a fall with better accuracy. Furthermore, Vaidehi et al. (2011); Gasparrini et al. (2014); Zhang et al. (2012) emphasised the use of depth image for a major reason that is to maintain the privacy of the person during monitoring. It is not possible to recognise a person observing a depth image captured by the depth camera and hence the identity of the person remains confidential. Zhang et al. (2012) proposed an automatic fall detection for elderly using RGBD camera which can maintain privacy making use of depth information and deploy RGB to support detection in case of the distance limitation of depth images. Their detection system recognised five classes of activities classified by Support Vector Machine (SVM) using kinematic information. The first stage of this study is partially similar to Zhang et al. (2012) where the classification of six activities are performed using Convolutional Neural Network. Tracking of body or body parts have been explored and applied in the past for fall detection. However, the pose estimation strategy has never been applied in fall detection to my knowledge. Some of the traditional tracking methods include particle filter and Kalman filter. Nait-Charif and McKenna (2004) proposed an overhead tracking strategy to recognise fall in inactivity regions such as the floor using a particle filter. Jang et al. (2002) proposed 2D human body tracking with a structural Kalman filter which utilizes the relational information among sub-regions of a moving object. The model uses previous time frame sub-regions information to define sub-regions in the current frame. Similarly, Chua et al. (2015) also uses the strategy to divide a human body into three sub-regions and detect fall by analysing the shape of the human silhouette achieved after background subtraction. The foreground blob is computed and then is divided into three portions with a ratio of 30:40:30 percent. Most of the tracking strategy is performed with different preliminary conditions which can suffer when they are not met. Some more cases where human body parts were tracked to differentiate fall with other normal activities such as standing and sitting were proposed by Bian et al. (2015); Rougier et al. (2006); Yu et al. (2009).Min et al. (2019) proposed human fall detection using normalised shape aspect ratio (NSAR) which is computed by dividing shape aspect ratio (SAR) by calibrated shape aspect ratio (CSAR). They mention that the shape aspect ratio changes substantially on camera layout variation which can be considered as an important clue to identify fall. The value of NSAR is close to 1 during walking and is largely different to 1 during fall. Fan et al. (2019) proposed fall detection using slow features which are extracted from the shape feature sequences achieved after background subtraction. The slow features contain discriminative information about human actions. They compute squared first order temporal derivatives of the slow features and use them for classification using SVM. Alzahrani et al. (2019) presented evaluation of skeleton features achieved from Microsoft Kinect V2 to analyse fall using 4 supervised learning techniques: Random forest (RF) decision tree, neighbourhood component analysis (NCA), artificial neural network (ANN) and support vector machine (SVM). Random forest was found to be the best performing classifier in this case.

2.2 Inactivity Detection Based Fall:

According to Delahoz and Labrador (2014), falls exhibit unique patterns and characteristics that can be exploited to detect and predict them. Characteristics such as the increase in negative acceleration of the body due to sudden change in position, falls ending with an inactivity period on the floor and sudden changes in height, width and inclination angle of the body are explored in the literature and have been used as a major clues for fall detection purpose as presented by Vaidehi et al. (2011); Yu (2008); Rougier et al. (2011). Iazzi et al. (2018) proposed a machine learning approach to detect fall based on posture analysis using SVM. They used background subtraction to extract human silhouette initially using the CodeBook model. However, they were not able to get satisfactory human blob out of the background subtraction method in the presence of shadows and the moving of furniture. In an attempt to achieve a clearer human silhouette, they detected shadows using HSV colour and gradient information and removed the shadows pixels using a pre-defined threshold. To determine the actual human silhouette only, they used two other assumptions: 1) remove all blobs which has an area smaller than 50 pixels and 2) classify other blobs into many classes by using the rectangle distance. The human silhouette was then determined by using the motion information based on optical flow and the distance between the current and previous positions of each class. Blob possessing higher motion and smaller distance was considered as the actual blob. Finally, they used histogram features from the rectangular bounding box of the human silhouette to classify four postures: lying, sitting, standing and bending using a Multi-class SVM. Once a lying pose is accurately classified, fall verification stage is triggered. In the verification stage, they look into next 10 sequential frames and expect to get all lying pose if its a fall. Their fall detection is based on the assumption that the fallen person stays immobile after fall. Jansen and Deklerck (2006) also investigated the area of the body and the orientation to analyse the fall using depth images. The change in orientation of the body is used to identify inactivity and if it exist in specified context, a fall is detected.

2.3 Head Motion Analysis Based Fall:

Nait-Charif and McKenna (2004) proposed a computer-vision based abnormal inactivity detection system in an indoor environment by tracking activities of the occupant in the different region of a room. They considered sitting on a chair or lying on a sofa as a normal activity occurring in the usual region and lying on the floor to be abnormal activity occurring in the unusual region. They used a particle filter algorithm to track movements of the head. However, it is debatable that a fall event can also end up lying on a sofa. The other weakness in the strategy is, a particle filter-based tracking can suffer as the acceleration of the body part is high in real time. During a fall, the acceleration is not predictable as each event occurs with different speeds. Moreover, the speed during the fall is normally greater than the speed during normal activity. This adds an extra challenge in tracking a fallen person. Halima et al. (2019) proposed a particle filter based head tracking strategy using the fusion of depth and thermal images. The paper suggested the fusion can improve accuracy in tracking the head silhouette which was obtained after background subtraction. Similarly, Rougier et al. (2013) proposed a method to extract ellipsoid shaped 3D head trajectory of a person using a single calibrated camera. Then the head was used for tracking during a fall based on particle filter that used color histogram and shape information.

2.4 Activity Recognition for Fall Detection using Machine Learning:

Ji et al. (2013); Simonyan and Zisserman (2014) proposed a convolutional neural network for human action recognition which extracts features from both the spatial and the temporal dimensions by capturing motion information and dense optical flow available from multiple adjacent frames. Similarly, Jain et al. (2013) also proposed a multi-layer convolutional network architecture that learns low-level features and a higher-level weak spatial model to perform human pose estimation. They found that training multiple stage CNN with one network per feature resulted in improved performance. Pose classification is the first and primary objective at this stage of our research work as this can give a clue about a lying pose after a fall event. The initial work is mainly inspired by the work of Zhang et al. (2012) which is using RGB-D images to recognise different pose and of Toshev and Szegedy (2014) that is pose estimation using the convolutional neural network. So far in the literature, RGB based data are mainly used in the convolutional neural network and therefore the combination of RGB and Depth was explored. The research aims to explore indoor based activities such as standing, sitting, lying, bending and crawling only in the first stage that could represent a fall when considered in a sequence. Hence, considering all these scenarios, the dataset that contains only these activities for training were not available. This situation motivated me to create my own data set. Although creating my own data set was going to be time-consuming and its accuracy in terms of labelling will bring a huge challenge, the data was created to specifically represent the major five poses mentioned above and make them more useful for this research. The data is now publicly available for similar academic research purpose. In past few years, machine learning techniques have become popular to differentiate fall from other normal activities. de Miguel et al. (2017) proposed vision-based fall detection approach where they combined several algorithms like background subtraction, Kalman filter, and optical flow to achieve input features for a machine learning algorithm. This approach can suffer in speed as different algorithms are combined which brings complexity to the approach.Panahi and Ghods (2018)proposed a fall detection system based on silhouette feature analysis after background subtraction using Support Vector Machine (SVM) and a distance threshold that considers the center of the human silhouette from the floor. They assumed that if the lying down on the floor is for a longer period of time, they consider it as fall. However, there are several chances of failure in this approach. They consider the largest contour after subtraction as the human object which can be misleading in case a pet is moving in the room. Other than that, lighting variations, limited depth distance and falls that does not end up on the floor are major issues in this approach.

Iazzi et al. (2018) proposed a machine learning approach to detect fall based on posture analysis using SVM. They used background subtraction to extract human silhouette initially using the CodeBook model. However, they were not able to get satisfactory human blob out of the background subtraction method in the presence of shadows and the moving of furniture. In an attempt to achieve a clearer human silhouette, they detected shadows using HSV colour and gradient information and removed the shadows pixels using a pre-defined threshold. To determine the actual human silhouette only, they used two other assumptions: 1) remove all blobs which has an area smaller than 50 pixels and 2) classify other blobs into many classes by using the rectangle distance. The human silhouette was then determined by using the motion information based on optical flow and the distance between the current and previous positions of each class. Blob possessing higher motion and smaller distance was considered as the actual blob. Finally, they used histogram features from the rectangular bounding box of the human silhouette to classify four postures: lying, sitting, standing and bending using a Multi-class SVM. Once a lying pose is accurately classified, fall verification stage is triggered. In the verification stage, they look into next 10 sequential frames and expect to get all lying pose if its a fall. Their fall detection is based on the assumption that the fallen person stays immobile after fall.

Doulamis and Doulamis (2018) proposed an adaptive deep learning approach to detect fall. They used deep learning to distinguish humans in the foreground from the background and use adaptive learning when there is a change in the background to retrain the model to confidently achieve human in the foreground. The model automatically triggers an adaptable mode when it understands that there is a significant change in the background of the current environment. This adaptability is possible by adjusting the weight constraints. The network weight adaptation is performed with an aim that only minimum degradation of previous knowledge is compromised so that the background data from the current environment is trustworthy as much as possible.

Similarly, Núñez-Marcos et al. (2017) proposed convolutional neural networks-based fall detection where they classified fall from non-fall activities using the transfer learning technique. They used the pre-trained VGG-16 CNN model on Imagenet with the stack of optical flow images as the input to the CNN model. Optical images represent the motion between two consecutive frames. However, this motion information is too short-timed to represent a fall and hence they used a stack size of 20 sets of optical flow images as the inputs to the CNN classifier. They applied the fine-tuning technique to the pre-trained VGG-16 model using UCF101 dataset from Soomro et al. (2012) to help the network to learn motion features. The fine-tuning technique is used to specifically narrow down the classification capability of a model from a higher number of classes to fewer classes. Although the model at this stage starts to recognise only fewer classes, the classification capability of the model on particular classes increases in comparison to the previously trained stage. They froze the weight from the layer which has the capacity to learn more generic features. Finally, they then further fine-tuned only last two layers of the model for the classification of two classes: fall or no-fall.

Another deep learning approach was explored by Shojaei-Hashemi et al. (2018) where they used 3D joint skeleton features achieved from Microsoft Kinect SDK to feed into an LSTM to identify fall. They have mentioned that fall samples are limited in comparison to other normal activities samples like walking. Therefore, they first train a multi-class LSTM on larger samples of human regular actions and transfer the learned weight to retrain only the last layer of a two-class LSTM. They used the transfer learning technique to avoid the need for a huge fall dataset for training on LSTMs which have similar structures until the second last layer. The last layer of the two LSTMs differs in the number of hidden units which is set according to the number of classes. They were able to achieve a precision value of 0.9323 and recall value of 0.9612 on the NTU RGB+D Action Recognition Dataset Shahroudy et al. (2016) which contains 56,880 video samples with 60 different actions including fall.

Solbach and Tsotsos (2017) proposed a vision based fallen person detection approach for the elderly which combines depth information with 2D human pose estimation based on CNN to estimate 3D human key points. These 3D key points are then used to achieve CoG (Center of Gravity) of the detected key points and UbC (Upper body Critical). Using the Euclidean distance, the distance between two derived points and the 3D points of the ground plane are then measured. The person is considered to be fallen only when either the distance with CoG or UbC is lower by 0.7m. The threshold 0.7m was chosen on the basis of information that the lying or seating arrangements are usually made higher than 0.7m for elderly. Their approach was able to achieve a true positive rate of 0.933 at home environment and 0.912 at the office environment. However, the approach heavily relies on ground plane detection which can affect the accuracy when no depth information is available which may simply be because of reflections. They have created their own dataset to analyse fall under home environment and office environment.

Wang et al. (2016) proposed 2 stage fall detection approach based on PCANet which was trained to predict the label for each image after background subtraction. Then they used SVM to recognise fall or no fall using the sequence of labels predicted from PCANet.

Chapter 3

Research Methodology

3.1 Problem Statement

To develop a fall detection system, one needs to address three major problems: (1) Classification of poses, (2) Track the person during all the possible postures and (3) Distinguish fall from other normal activities.

3.2 Approach

This study purpose a fall detection system in four stages. These are as follows: a) Human silhouette extraction, b) Activities recognition, c) Fallen person posture detection and d)Classification of fall on the basis of the detected postures.

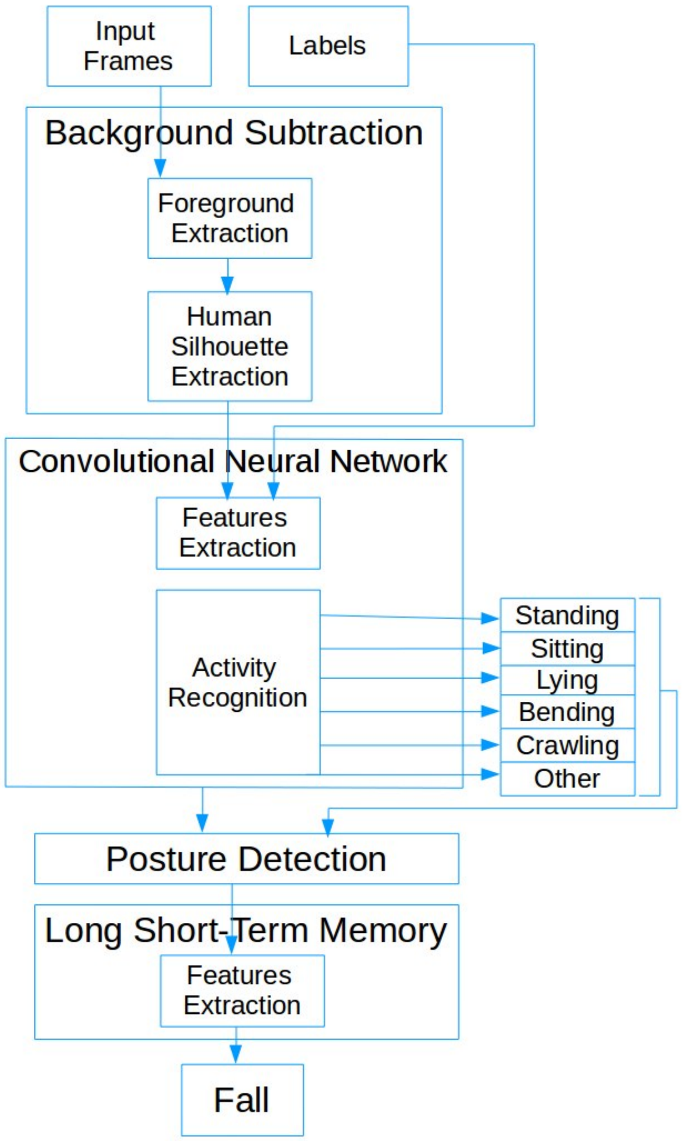


Figure 3.1: *Conceptual Block Diagram.*

Figure 3.1 represents the conceptual block diagram of the proposed approach to detect fall. The conceptual block diagram can be explained as:

- The input block represents images that are recorded from a single un-calibrated Kinect sensor which is mounted at ceiling height.

- The label block contains information about the pose which represents the different type of activities in each image.

- The background subtraction block in the conceptual diagram is used for extraction of a human silhouette. The object of interest is obtained in the foreground image by subtracting the background of the input image. Although background subtraction is used for human silhouette detection, it can suffer when there is another moving object is present together with the person. The other moving object beside the person can simply be another person, a toy, pet, a support stick or a fan. This is one of the drawbacks of this approach as it produces noise in the foreground image after subtraction of the background. Moving objects can also overlap and have deformation issues as the silhouette may not appear to look like a human visually after the overlap. To avoid this issue, the input images are recorded carefully to ensure that the only moving object in the image is human for classification of poses. Even then there can be a slight change in the background image during the movement of the person. This will eventually produce further noise in the input image. The reason for considering only a single person is that, if a fall occurs in the presence of another person, the other person can easily support the fallen person. This is therefore mentioned as one of the limitations of our approach. The volume of the dataset is the key to solve this issue using CNN. As long as there are plenty of cleaner images where the person does not overlap, the model is robust enough to learn the features from these images and ignore several variations and challenges that exist in the fewer number of datasets. Datasets are therefore vital for a CNN model as their accuracy can easily get tampered if there are plenty of disturbed or noisy images in the training set. This stage is discussed later in section 3.3.1.

- After the background subtraction stage, the input image is then fed to a CNN. The network is able to learn generic features from the training images and able to recognise similar features for a new set of images that it has never seen before. The CNN model is trained to learn different poses represented by the input images and expected to classify the unseen images during the test. For evaluation, the true labels are compared with the prediction made by the classifier. The CNN model needs to learn the posses from the labelled training images. This approach can also be referred to as supervised learning. In supervised learning, the model is supervised to learn certain features based on the labelled inputs and is ultimately expected to predict similar label in unseen data after the training period Kotsiantis et al. (2007). The working and architecture of the convolutional neural network are discussed in detail in section 3.6.

- After pose recognition, pose estimation is performed using the capability of CNN again. The idea behind using the pose estimation is to acquire stable posture location information that can be used as features to distinguish fall from other normal activities.

- The posture location information is then fed to Long Short-Term Memory(LSTM) recurrent neural network. LSTM is capable of learning order dependence information and is, therefore, able to tackle the prediction problem in sequential data. They have an internal state that can use context information learnt from previous time steps to influence the prediction at the current time stepsSak et al. (2014). A fall is a special case of sequential change of poses where the changes are at a higher rate in comparison to normal activities. Hence, this higher rate of change is a unique feature that distinguishes fall from other activities. In this study, the posture detection is performed to acquire the location information of the person. One can further analyse the rate of change of postures in a series of frames from the location information. This information is crucial and can be used as features in the LSTM network to classify fall and other normal activities.

3.3 Relevant Techniques

3.3.1 Background Subtraction

The first step for the fall detection consists of detecting human silhouette which is possible with the help of background subtraction. Using this technique, it is possible to obtain a human silhouette in the foreground after subtracting the current image from a reference image. As illustrated by OPENCV (2014), if C represents the current image and B as the static background image, then an image pixel is said to be foreground F if

$$F = \begin{cases} \text{C}, & if\,|C - B| > \tau \\ 0, & \text{otherwise} \end{cases} \tag{3.1}$$

where τ is a predefined threshold.

The proposed fall detection is based in an indoor environment where the possibility of a change of scene in the background is minimal and hence static background is considered. The foreground obtained after background subtraction is used for feature extraction. The human silhouette had some noticeable noise due to the certain variation of light. This indicates the fact that light plays a significant role in background subtraction. However, depth images are less affected by lighting changes. Furthermore, depth images also come with its limitation of distance. Stone and Skubic (2015) mentioned that a fall detection model can suffer due to the curse of distance limitation and noise if only depth input is considered. The depth is estimated only up to a limited distance (typically less than 5m) together with the inclusion of noise and their field of view (aprox. 60 Degree) is also far more constrained than that of the specialised camera (approx. 180 Degree) Zhang et al. (2012); Henry et al. (2012). Combining RGB and depth is possibly a suitable solution as they can complement each other.

3.3.2 Convolutional Neural Network

Overview

Convolutional Networks are biologically inspired trainable architecture composed of filter banks, some non-linearity and feature pooling layers that are capable of learning invariant features that could help in the classification LeCun et al. (2010). These invariant features represent those general characteristics that exhibit unique relevant information about the object of interest despite different variations. The CNNs are also data driven approaches similar to ordinary neural networks as they rely heavily on the training dataset of labelled images. A high amount of data is necessary to achieve variations in the training set that reflect what could exist in the target environment.

In general, the convolution of an image refers to an operation to analyse signal or features within an image after the effect of a filter or kernel. A convolution is done by multiplying a pixel and its neighbouring pixels colour value by a matrix of numbers called filter. It works by determining the value of a central pixel by adding the weighted values of all its neighbours together. The filter moves across the image from left to right in a step by step manner in search of features in the image. Filters can be of different sizes and can contain a different pattern of numbers to produce different features in the image Skymind (2016); Ludwig (2015).

Classification Approach with CNN

In computer vision, the problem of activities recognition can also be considered as image classification problem. Different poses can be uniquely classified and recognised comparatively from the perspective of a human eye. Therefore, the fall detection system can be taken a step further by attempting to recognise the activities in terms of five different poses and considering them as five different classes: **class 1. Standing**, **class 2. Sitting**, **class 3. Lying**, **class 4. Bending**, and **class 5. Crawling**.

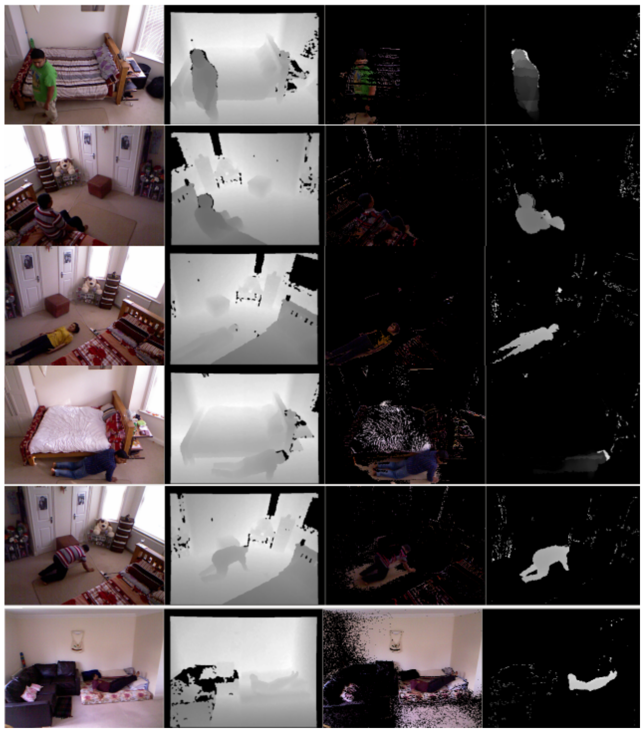


Figure 3.2: *Different Poses in training dataset: a. RGB, b. Depth, c. Background subtraction in RGB, and d. Background subtraction in Depth*

.

In the above figure 3.2, different types of poses that are available in the dataset are illustrated in four different forms of images: RGB, Depth, background subtracted RGB and background subtracted Depth. These poses are classified as different classes and are labelled as ground truth. Apart from the five different classes mentioned earlier, there is a special case where the occupant is absent from the room. Therefore, the system should also understand that the occupant is absent. A totally occluded case can also be considered as this special case where an occupant is absent. Such scenarios are categorised into a different class by labelling them as 'Other'. One of the special abilities of the supervised learning based model is to learn from the training data. Therefore, it is important to include such special cases in the training data so that the model can classify such cases later in the validation and test data.

Regression Approach with CNN

The classification approach does not require to predict real or exact values and can just have generic values that represent a particular class. However, there are cases that would require to predict real values. Pose estimation is a classic example of a regression approach. A number of joints are identified with their exact location information in the image. It is expected from the training that a neural network is able to make similar predictions in an unseen image.

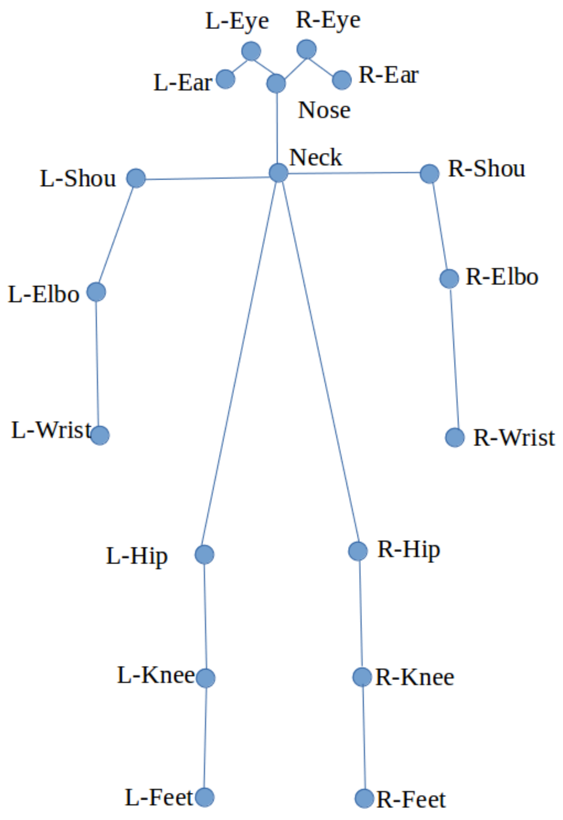


Figure 3.3: *Human joint estimation*

.

As seen in the above figure 3.3, there are several numbers of joints that are labelled with their exact location information. These locations of joints are used as ground truth to train a neural network and expected to predict joints locations in unseen images. There are plenty of applications for this kind of joints predictions. Location of joints can provide an important clue regarding the location of a person and can be used in tracking the person. Generally, a regression problem is considered to be tougher than classification as the model need greater precision in prediction of exact values to achieve a solution.

Next section will illustrate the complete architecture of convolutional neural network including its operations in different stages.

CNN Architecture

Convolutional neural network architecture is made up of three major layers between input and output. They are illustrated in figure 3.4: Convolutional Layer, Pooling Layer and Fully Connected Layer. There may be few repetitions of these layers before the final output. Increasing the number of layers makes the network deep which in turn can help to acquire further complex features of the input LeCun et al. (2010).

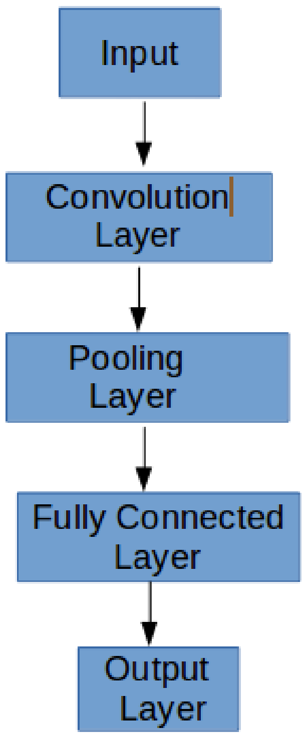


Figure 3.4: *General CNN Architecture*

.

Input

The first part of Convolutional Neural Network architecture is the input layer where the network receives images as the major inputs. According to Skymind (2016), the image received in the input layer of the convolutional neural network is considered as the volume of the n-dimensional object measured by width and height.

Convolution Layer

As shown in the above figure 3.4, the next block represents the convolution layer which undergoes three major operations: convolution, spatial batch normalisation and Rectified Linear Unit (ReLU).

Operation1: Convolution

From the input image, an image patch is randomly cropped from the entire image. Later this patch is used to perform convolution with a number of filters of a certain size. The size of this image is chosen in such a way that during the convolution operation, a filter of a certain size can operate over the entire image with minimum risk of losing pixels in the boundary. Padding is applied at the boundary of the image so that the filter fits well with image patch and minimises the risk of missing pixels at the boundary.

This layer of the CNN is responsible for generating the features maps at different locations of an image by the process of convolution with the filters(kernels).

A kernel looks for the same features but at a different location of the input image.

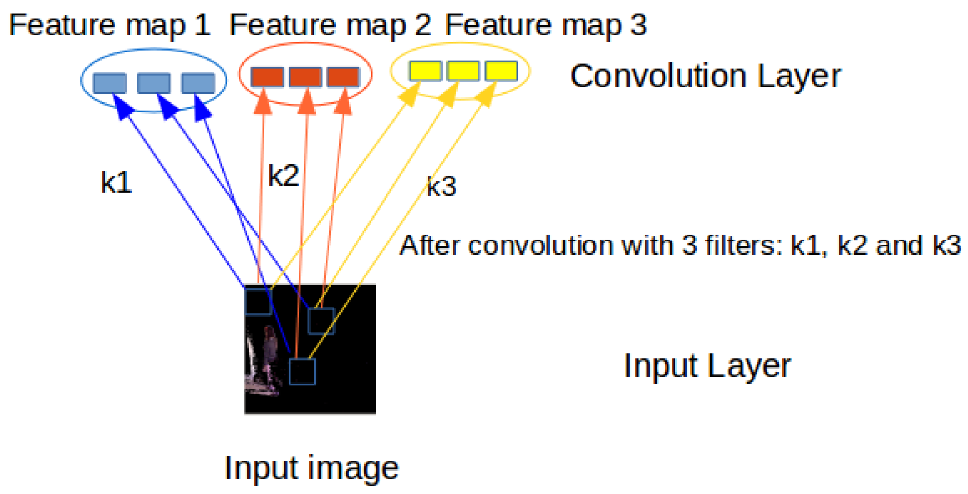


Figure 3.5: *Feature maps formation after convolution*

.

Figure 3.5 illustrates the formation of three feature maps from the convolution operation with three filters. Here in the figure 3.5, similar features share weights and are placed together as a single feature map. Three different colour represents three types of features obtained from three filters at three different locations of the input image. These features are local pixels which are highly correlated and can form edges that are picked in the initial stage of the convolution. These correlated pixels transform into a unique feature. Let x be a colour image of size m_2 x m_3 and x_i be feature map of each input. Let y be the output feature map composed of n_1 feature maps of size n_2 x n_3 and each output feature map is denoted by y_j. Let $K_{\{i,j\}}$ be the trainable kernel of size l_1 x l_2 linking $i-$th input map to $j-$th output map. The output feature map can then be computed as in LeCun et al. (2010):

$$y_j = b_j + \sum_i k_{ij} * x_i \tag{3.2}$$

where $*$ is 2D discrete convolution operator and b_j is a bias parameter.

Operation2: SpatialBatchNormalization

The other operation that is processed after convolution is spatial batch normalisation. For normalisation, mean and standard deviation are calculated from a mini-batch of inputs as a pre-process. After convolution operation, all the features from each feature map are then subtracted by the calculated mean and divided by the calculated standard deviation(σ). It is also referred to as SpatialBatchNormalization Ioffe and Szegedy (2015).

$$y = \frac{x - Mean(x)}{StandardDeviation(\sigma) + Constant(\beta)} \tag{3.3}$$

Subtracting each feature with the mean is done to centre the data around the origin and once the data are zero-centred, they are then normalised by dividing by standard deviation. A small value of constant (β) is added to standard deviation term to avoid divide by zero cases. In this case, beta (β) is set to 1e-3. The only reason for this normalisation is that it is possible that different input features have different scales and therefore they should be scaled to the same level so that they become approximately equal importance for learning Li and Karpathy (2015). In other words, consistency in the features is very crucial.

Operation3: ReLU

ReLUs are used to transform an input to a different domain such as from linearity to non-linearity. They are also known as activation functions in general. The network needs to achieve distinguished properties of the input to learn. Since the input features are linear in nature, they are not easily distinguishable in the case of only a small change in weights in the input value. Hence an activation function serves as a threshold to generate non-linearity from the linear input. Non-linearity is applied to the inputs to separate high-level and low-level features from the input.

ReLU maps the input x to max(0,x) which means that it maps negative inputs to zero and positive inputs become the outputs without any change. Simply, the output x after ReLU operation means features are present and zero means that there are no features. Similarly, some other activation functions that are used in neural networks are: The sigmoid function which maps an input to a value in the range from 0 to 1 and Tanh function which maps an input to a value in the range -1 to 1.

Pooling Layer

The final attempt for the selection of features from a pool of features available in the activation map is performed in the pooling layer. The activation map is the feature map which is achieved after the ReLU operation. Since only high-level features are selected from a region, the operation is called max-pooling operation.

MaxPooling Operation

The max-pooling is the method of selecting the most responsive node from a region of an activation map. A region of a certain size is selected for pooling and a certain stride is considered assuring the selected area do not overlap while passing over the entire activation map. The selection of size is also done considering the region fits best over the activation map so that it does not lose any information near the boundary. Once the size of the region and stride are determined, the largest value from the region of interest is selected and the rest of the values are discarded from the activation map. This results in a reduced resolution output feature map since weaker information is left out. However, these features are more superior and sensitive to small variations but they do not precisely tell where these features were located in the previous layer LeCun et al. (2010). The two tables below demonstrate the max pooling operation as demonstrated in Li and Karpathy (2015).

1	1	2	4
5	**6**	7	**8**
3	2	1	0
1	**5**	3	**4**

Table 3.1: *Single feature map*

— Maxpooling(R=2x2,S=2x2) $\succ$

6	8
5	4

Table 3.2: *Max-pooled feature map*

Table 3.1 shows single activation map of size 4 x 4 which undergoes a Max-pooling operation with a region (R) of width = 2px and height = 2px and the next batch of selection takes place at a gap or stride (S) of 2. Here the 2x2 square region can move through 4 other blocks within the activation map. Hence it will produce 4 maximum values from the pool of 4 regions and preserve it in the same order as they were taken. The maximum value is shown in bold in table 3.1 will be selected. Table 3.2 represents the maximum values pooled out of the activation map and the overall 4 x 4 size of the activation map is down-sampled to exactly half producing the output activation map of size is 2 x 2.

Fully-connected Layer

The features, after passing through several convolutions and pooling layers are stacked together finally in the fully connected layer. All these features from the previous layers are connected to the output with the help of the fully connected layer (FC). A fully connected layer is linearly connected to the output. However, it allows a non-linear combination of these features to extract more complex features using more than one fully connected layer. A number of features are carefully selected on the basis of observation assuring a significant number of features are considered for recognition and at the same time, these number of features does not lead to over-fitting. Over-fitting is the condition of the network when it has higher capacity than required leading to poor performance. Further detail on over-fitting is discussed later in the chapter.

Output Layer

The last layer is the output layer which predicts the output class. However, the output layer uses a classifier which computes the probability of the class score which is achieved from the fully connected layer. Finally, the maximum value out of these probabilities represents the corresponding output class label. The softmax is a special classifier used in the output layer of the convolutional neural network that computes the exponential of these scores and then divides each of these values by the sum to normalise and achieve a uniform distribution of probabilities that sum to one. A negative log is then applied to these normalised probabilities to calculate cross-entropy loss. Therefore, the log of softmax which is also referred to as LogSoftMax encourages the normalised log probability of the correct class to be as high as 1 Li and Karpathy (2015).

3.3.3 Simple Recurrent Neural Network

Background

Recurrent Neural Networks (RNN) or Simple RNN came into existence when there was a need for understanding of contextual information from a sequential input related problems. Information is correlated in sequential inputs. The temporal information that is acquired from the connection of sequential data is very critical for this kind of network.

The information passed in the previous stage of RNN can influence the outcome in the present stage. This can be further clarified with an example of describing a fall scene. If one wants to recognise a fall event in the video using the RNN, the RNN model can relate a lying pose in the current frame to be a part of a fall. It can understand this by referring to a few previous frames where the person could be standing or sitting. The same thing applies again in a situation where a lying pose is observed in the current frame and when referred to the previous frames, the model could find more lying poses. In that sense, the model might understand the whole scenario as a non-fall case or simply a lying case only.

RNN Applications

RNN model can have variable input and output length. This features can be used for different applications. As shown in figure 3.6 from Karpathy (2015), there are four applicable scenarios where their performance are widely tested and their possibilities are immensely explored.

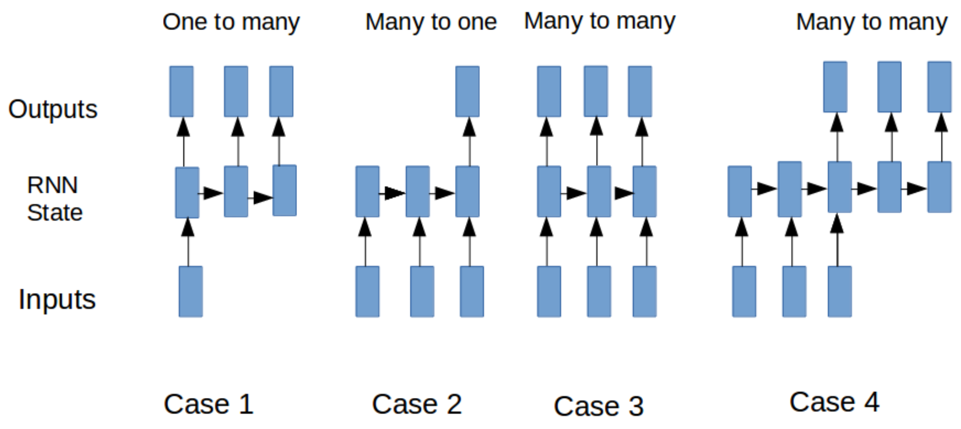


Figure 3.6: *Input and output vector arrangements applicable in a RNN model from Karpathy (2015).*

- **Case 1:** The first case is the scenario where a single input is passed to the RNN model and expected to predict output in the form of sequence. An example of such type of case is used for image captioning as proposed by You et al. (2016). The RNN model takes an image and provides a label that consists of words to make a sensible sentence.

- **Case 2:** The second case is the scenario where the sequence of inputs are passed to the RNN model and expected to predict a single output. An example of such type of case is used for sentiment analysis as proposed in the paper Zhang et al. (2016). The model takes sentences as input sequences and classifies it as a positive or negative sentiment. Another example of such cases is to feed a sequence of images to recognise an action using such models.
- **Case 3:** The third case is the scenario where the synced sequence of inputs are passed to the RNN model and expected to predict synced output sequences. An example of such type of case is used for labelling of each frame to understand human activity in a video in more details like proposed in Yeung et al. (2018).
- **Case 4:** The fourth case is the scenario where the sequence of inputs are passed to the RNN model and expected to predict some output sequences. An example of such type of case is used for machine-based translation work. A model is able to recognise the words that are in English and then able to output the words with similar meaning in French as proposed in Bahdanau et al. (2014).

Backpropagation Through Time

The RNN contains a loop set-up where information is passed from one step of the network to the next and the model learns to relate the things as a pattern from the longer sequences. It can be considered as multiple copies of the same network interconnected in the activation state. The inter-connection allows the passing of information from the previous time step to the current time steps.

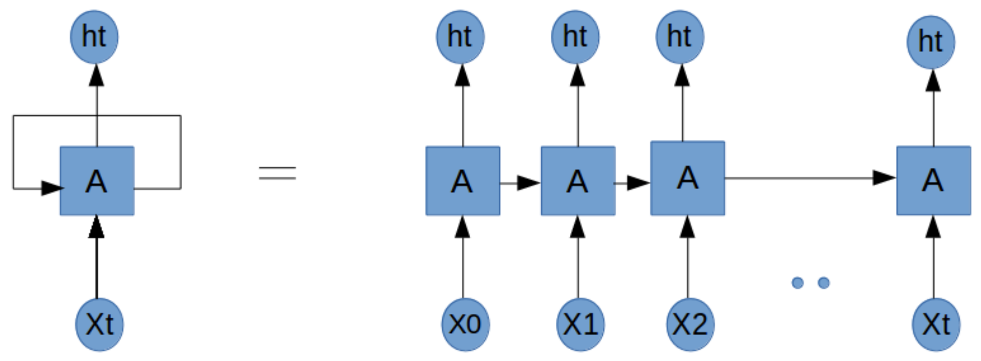


Figure 3.7: *Unrolling of an RNN model into different time-steps from Olah (2015).*

An example of an RNN having a loop can be seen on the left-hand side of the equation in figure 3.7. An unrolled RNN structure is illustrated on the right-hand side of the equation. The chain-like structure that is seen in their architecture is used to pass the message to their successor at different time-steps. Recalling the backpropagation theory from the earlier chapter for CNN, they are basically used to tweak the weights of the model during training to achieve the minimal error by going back through the network layers. In the process, the partial derivatives of the error with respect to weights are obtained. This is then used by gradient descent algorithm to adjust the weights so that the error between ground truth and the prediction decreases further. However, in the case of an unrolled RNN, backpropagation is done on the basis of time-steps and hence is called Backpropagation Through Time (BPTT). In the case of RNN, the error of given time-steps depends on the previous time-steps and therefore the error is calculated at each time step to update the weights. Hence, these arrangements can be computationally expensive and time-consuming when there is a higher number of time steps. The major issues are caused due to the gradients update that takes place at each time steps during backpropagation.

Issues in Recurrent Neural Networks

RNN suffers therefore with two important problems: Exploding Gradients and Vanishing Gradients.

- **Exploding Gradients:** Exploding Gradients is the case when the value of the gradient is too high. The size of the gradients build up further at each time steps by accumulation during each update and eventually explode. This results in poor prediction due to an unstable networkPascanu et al. (2012). Gradient clipping is a method to limit the value of gradient going higher than a norm with a certain predefined value. The other solution to this problem is to use regularization and dropout technique to improve the performance of the model Zaremba et al. (2014).
- **Vanishing Gradients:** Vanishing Gradients is the case when the gradients accumulated over the time steps start to get too low or become zero. This can lead to little to no training of the model. Gradient clipping and Dropout can be used as the solution to tackle this problemPascanu et al. (2013). Furthermore, using ReLUs as an activation function which represents the activation function to have maximum value only between 0 and the input. This restricts the gradient value going lower than zero and provide a small gradient to give a chance for the model to keep learning Talathi and Vartak (2015).

RNNs are able to use information from the previous state, understand the context and determine the future output. RNN has been successfully deployed in a situation where the relevant information within a short time step is enough to understand the context for an input. There are some cases where longer time steps are necessary to describe a context in a better way. In case of long time steps or longer input dependencies, RNN can forget the context of what has happened in a previous step. RNNs are unable to learn to connect the information when the chain is longer. The internal structure of the repetitive chain of state shown in figure 3.8 is discussed below to understand the working of RNN further.

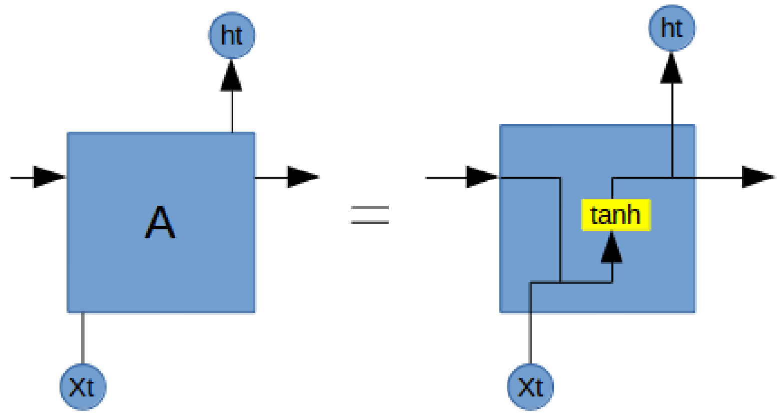


Figure 3.8: *Internal structure of a simple RNN state Olah (2015).*

A simple RNN passes the information from the previous input through a single tanh layer to the current time-steps. That means the gradient can have value lesser than 0 which is the major reason for the vanishing gradient problem.These issues are solved by another type of recurrent neural network called the Long Short-Term Memory(LSTM).

3.3.4 Long Short-Term Memory Networks

Background

LSTMs are a special kind of RNN that are capable of learning long-term contextual information. They are specially designed with internal memory state to overcome long-term dependency problem observed in simple RNN. LSTMs enables RNNs to remember longer dependencies information with the help of the memory units where they are also able to read, write, store and delete this information. These memory units are categorised into three gated operational cells: input gate, forget gate and output gate. An RNN network becomes LSTM network when the units of an LSTM are used as the building units of the layers of RNN Phi (2019).

LSTM Architecture

A repeating chain of state of LSTM architecture is shown in figure 3.9 below.

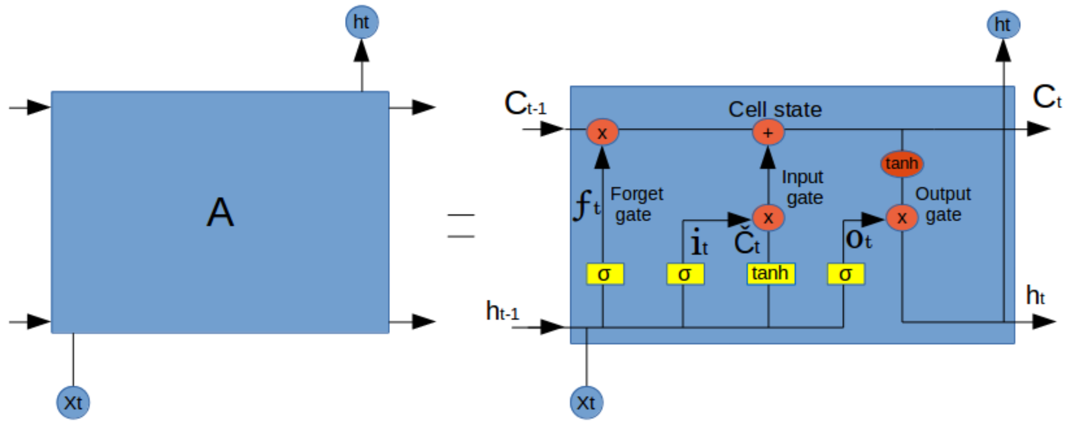


Figure 3.9: *Internal structure of a LSTM state Olah (2015).*

In figure 3.9, the arrow line indicates the transfer of the output vector from one node to the input of another. The orange circles are point-wise operations such as vector addition and the yellow boxes are the learned neural network layers. 'C' represents the cell state and 'h' represents the input or output state at current or previous time-steps. The three gates in the LSTM structure decides what to do with the contextual information based on the importance it assigns to the information with the help of learned weight Sak et al. (2014). The features of these three gates are:

- **Forget gate:** The forget gate denoted by f$_t$ determines whether to keep the information or delete based on its importance. It uses a sigmoid layer called as 'forget gate layer' which outputs 0 or 1 in the cell state C$_{t-1}$. O indicates that the information is completely deleted and 1 indicates that the information is completed stored based on previous output h$_{t-1}$ and input x$_t$. This can be expressed mathematically as in Pascanu et al. (2012):

  $$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{3.4}$$

  item **Input gate:** The input gate denoted by i$_t$ determines which new information to store in the cell state. It has two parts. First, a sigmoid layer called an 'input gate layer' which values will be updated. Second, a tanh layer creates a vector of new candidate information values given by $\check{C}_t$. These two things are then added to update the cell state. This can be expressed mathematically as in Pascanu et al. (2012):

  $$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i) \tag{3.5}$$

  $$\check{C}_t = tanh(W_c.[h_{t-1}, x_t] + b_c) \tag{3.6}$$

  Once these things are done, it is now possible to update the previous cell state C$t-1$ with C$_t$. The old cell state is multiplied by f$_t$ and then i$_t$*$\check{C}_t$ is added. This gives the new candidate value that will be used for the update. This update can be expressed mathematically as:

$$C_t = f_t * C_{t-1} + i_t * \check{C}_t \tag{3.7}$$

- **Output gate:** The output gate denoted by O$_t$ lets the information to influence the output at the current time-step. For making the output decision, there are another two steps. First, a sigmoid layer is used to decide what parts of the cell state are taking part to produce the output. Second, the current cell is passed through the tanh to restrict the values between -1 and 1. This is then multiplied with the output from the sigmoid state to obtain the desired output. This can be expressed mathematically as:

$$O_t = \sigma(W_o.[h_{t-1}, x_t] + b_o) \tag{3.8}$$

$$h_t = o_t * tanh(c_t) \tag{3.9}$$

3.4 Summary

This chapter presents the actual problem in this research and proposes an approach to tackle this problem. The conceptual block diagram of the proposed model is presented and discussed in detail. Furthermore, relevant techniques necessary to understand the overall working of the proposed model are discussed. These can support the proposed approach with background information and justify the reason for their selection.

Chapter 4

Activity Recognition Using Convolutional Neural Network

4.1 Proposed CNN Model

The figure shown in 4.1 below illustrates the architecture of the proposed CNN model. The design and the necessary settings are also discussed in details to further understanding the working of CNN. The relevant theories and principles necessary for CNN to operate successfully are also discussed in detail.

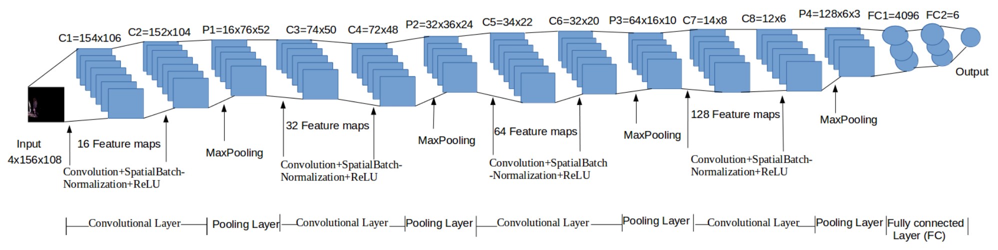


Figure 4.1: *Proposed CNN Architecture*

.

The input image is of size 240x360. An image patch of size 156x108 and filter of size 3x3 are selected. Each dimension of the image is stacked separately one on top of the other forming a depth volume of n-layers which is also referred to as channels. In this case, RGB and Depth images are used as inputs, and therefore, the input plane has four-dimensions altogether: three from RGB and one from the depth. Hence, the input volume as seen by the input layer is equal to a 4 x 156 x 108. The second layer in CNN is the convolution layer. In the architecture, C1 and C2 represent the first and second stages of convolution which generates 16 features map that is extracted from 16 filters at a stride of 1 x 1 and zero padding. The blue slices in stage C1 and C2 stages in figure 4.1 should represent 16 feature maps generated from 16 filters although the right number is not displayed in the illustrated figure. However, the number of feature maps have been mentioned correctly in different stages of the architecture in the CNN architecture.

Four hyper-parameters are essential to perform convolution. These are a number of filters(K), the size of the filter (Kw, Kh), stride (S) with which the filter will slide over the image and the amount of zero padding (P) in case the filter does not exactly fit within the image. Therefore the output volume can be represented as K x W2 x H2 as in Li and Karpathy (2015) where: W2 = output width=(input width(W1)-kernel width(Kw)+ 2 x P)/S + 1 , H2 = output height=(input height(H1)-kernel height(Kh)+ 2 x P)/S + 1 . In our case, we have S=1, P=0 and (Kw, Kh)=(3,3). Hence the output volume after the C1 stage is represented as 16 x 154 x106. Similarly, C2 operates receiving the output of C1 as input and hence the output of C2 is 16 x 152 x 104. During the max-pooling stage, the size of the image patch reduces to half from the previous convolution layer but with the same number of the filter. Therefore, after the first pooling stage (P1) the output is 16 x 76 x 52. There is a repetition of similar stages and operations to obtain 32, 64 and 128 feature maps respectively before the network introduces the fully connected layer.

After 8 stages of convolution(c1-c8) and 4 stages of pooling operation(p1-p4), there are two fully connected layers FC1 and FC2. In FC1 layer, 4096 features are considered. Similar, in FC2, only 6 non-linear combinations of features out of 4096 features are considered which also represents the class score (six different class).

4.2 Dataset

The datasets that are used for the simulation purpose are raw RGB and Depth images of size 640x480 recorded from a single uncalibrated Kinect sensor. The Kinect sensor is fixed at roof height of approx 2.4m. The datasets contain a total of 21499 images. Out of total datasets of 21499 images, 15800 images are used for training, 3199 images are used for validation and 2500 images are used for the test. The images in the dataset are recorded in five different rooms which consist of 8 different view angles. There are five different participants out of which there are two male participants of age 32 and 50 and three female participants of age 19, 28 and 40. All the activities of the participants represent five different categories of poses that are standing, sitting, lying, bending and crawling. Some images in the datasets are empty which are categorised as 'other'. Images of two participants: the male of age 32 and the female of age 28 combining total of 15800 images are used for training, and 3199 images for validation that contains a male participant of age 32 from the training set. The validation dataset images are recorded in a different room to that of training and testing set. Similarly, the test set contains images of three participants out of which two female participants are of age 19 and 40 and a male participant is of age 50. These images are also recorded in a different room that is not seen in the training or validation set. These total of 21499 images are in sequence but have not repeated anywhere in the sequence. Hence, this test set contains images recorded with unseen participants and unseen room against training and validation set.

Data augmentation is a technique that is used to increase the size of the dataset by transforming them into more varied forms with operations such as rescaling, flipping, cropping or rotating. This will not only help to increase the size of the dataset but also achieve variations in the dataset. This means that the class of the object will remain the same despite having different variations Wu et al. (2015). Furthermore, horizontal flip, scaling, cropping and rotation are also used as data augmentation technique to the dataset. This gives the model an opportunity to observe more data with different variations and improve by learning more important invariant features for classification. An image contains various information. However, depending upon the type of application, the region of interest (ROI) are identified. Then ROI can be considered as the primary deciding factor for influencing the learning of a model during training. Image cropping is a strategy in data augmentation which is basically used to remove unrelated information. This technique helps to enhance the overall visual perception of the image if the ROI is present within the image. This technique can contribute in two ways: First, the learning capacity of the model can improve and second, the processing speed of the model can also improve when dealing with lesser information. However, image cropping is only of an advantage when the ROI is available after the crop. There is a risk of losing essential information that could cost the visual perception of an image. Therefore, cropping is performed in an image before feeding to the model with careful consideration which is not losing the ROI. It is better to visualize the output image after cropping to make sure the ROI is clearly available within the image before feeding them to the model for training. The initial part of this research uses background subtraction solely with the view to get only the ROI or object of interest in the output image. All other information was irrelevant for describing a pose. However, it is not completely possible to get rid of all the irrelevant information after background subtraction. There can be some noises and some moving objects that will appear in the foreground or output image. Therefore, another attempt of getting more relevant information only was done with the help of image cropping.

Five different posses are classified as five different classes to form labels for all the dataset: These classes are categorised as class 1: Standing, class 2: Sitting, class 3: Lying, class 4: Bending and class 5: Crawling and class 6: Other which is also set as default for cases when the person is absent.

Computation is done on 3.7GHz Xenon HP Workstation Z420 with single Titan X 12GHz GPU and 12 GB of Ram using Lua Script and Torch7 library.

In the upcoming sections, the necessary strategies on training, testing and performance evaluation metrics of the model are explored before demonstrating the experimental results.

4.3 Training

4.3.1 Learning with Gradient Descent

A training session simply refers to learning by error correction. Let the desired target (Label) be $\mathbf{T}$ and the system output be $\mathbf{Y}$. The error $\mathbf{E}$ signal can then be computed as $\mathbf{E} = \mathbf{T} - \mathbf{Y}$. A ConvNet model aims to maximise the probability of correct prediction by minimising this error function. In gradient descent, the learning algorithm computes the derivative of the output with respect to the input and these derivatives are used to manipulate weights iteratively for every input so that the output has increased the probability of predicting the correct output. This also means gradient descent (GD) algorithm is used to minimise an error function by updating the weights positively or negatively. The correction of weights takes place in backwards and hence this method is called the back-propagation method. A cost function $\mathbf{E(w)}$, can be expressed in terms of the Mean Squared errors (MSE) as in Cauwenberghs (1993); Karnin (1990) as:

$$E(w) = \frac{1}{2}\sum_{i}\left(Target^{(i)} - Output^{(i)}\right)^{2} \tag{4.1}$$

And magnitude and direction of the weight update are computed by finding gradient or slope of the cost function by a step in its opposite direction which can be written as:

$$\Delta w_j = -\eta\frac{\partial E}{\partial w_j} \tag{4.2}$$

where $\boldsymbol{\eta}$ is the step size or the rate of convergence of weight. It plays a significant role in achieving global minima choosing the path of steepest descent. Therefore the weights are then updated after each epoch or iteration as shown in equation (4.3) below.

$$w := w + \Delta w \tag{4.3}$$

where $\boldsymbol{\Delta w}$ is a vector that contains the weight updates of each weight coefficient w.

The gradient associated with the weight can be positive as well as negative depending upon the position of initial weight. However in both the cases, gradient descent helps in convergence along a steepest path to reach global minima.

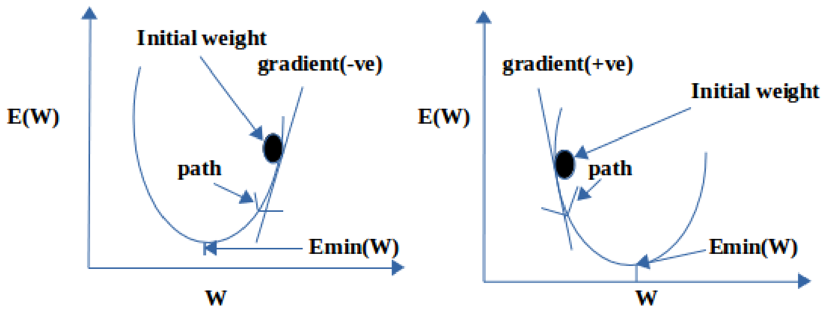


Figure 4.2: *(a)Gradient is negative (b) Gradient is positive. Ng (2015)*

If the initial weight is positioned to that shown in figure 4.2(a), using gradient descent the weight update will take place in opposite direction to weight axis to achieve global minima. Hence the tangent or slope of the gradient is negative and can be mathematically expressed as:

$$\frac{\partial E}{\partial w_j} \leq 0 \tag{4.4}$$

If the initial weight is positioned to that shown in figure 4.2(b), using gradient descent the weight update will take place in the positive direction to weight axis achieve global minima. Hence the tangent or slope of the gradient is positive and can be mathematically expressed as:

$$\frac{\partial E}{\partial w_j} \geq 0 \tag{4.5}$$

Now from equation equation(4.2) and equation(4.3), we can deduce the update in weight as:

$$w := w - \eta \frac{\partial E}{\partial w_j} \tag{4.6}$$

$\boldsymbol{\eta}$ is always a positive value and therefore, in both the cases of gradient being either negative or positive as in equation(4.4) and equation(4.5), the weight update in equation (4.6) will always converge towards global minima. Furthermore, it plays a significant role in the convergence of the gradient descent in two different ways:

Case1: Step-size too small: In this case if the $\boldsymbol{\eta}$ is too small, gradient descent can take a long time to converge which is not efficient.

Case2: Step-size too large: In this case if the $\boldsymbol{\eta}$ is too large, gradient descent algorithm might only oscillate and also diverge instead of converging to local minima.

In Gradient Descent (GD) optimization, the gradient is computed based on the complete training set and hence is also referred to batch gradient descent. However if we compute gradient as per training sample by not accumulating weight as in GD, then it is referred to as stochastic gradient descent. In case, the training samples are very large in number, computing GD can be costly in terms of convergence time. In every iteration, it runs through the complete training set, accumulates all the weight updates in a vector and then updated as shown in equation 4.6. Whereas Stochastic Gradient Descent (SGD) uses only one randomly shuffled training sample, computes gradient and updates weight simultaneously and therefore, convergence starts improving straight away from the first sample Bottou (2010).

Stochastic learning can also perform better in comparison to gradient descent due to the presence of noise in the updates by jumping randomly into another possibly deeper local minima. However, due to its stochastic nature, the path towards global minimum is fluctuating because of noisy approximation of the true gradient. Therefore, it may not successfully converge undergoing weight fluctuations. The size and the fluctuations depend on the degree of noise during the stochastic updates and the size of learning rate $\boldsymbol{\eta}$. One of the approach to reduce these weight imbalance is to use mini-batch gradient descent. A min batch gradient descent is a mid-way approach between gradient descent and stochastic gradient descent method of learning. In this approach, a small mini-batch size of training samples $\boldsymbol{K}$ is considered and instead of computing gradient for either 1 sample (SGD) or the complete $\boldsymbol{N}$ number of training samples (SD), the gradient is computed from $1 < \boldsymbol{k} < \boldsymbol{N}$. Therefore because of the frequent update in weights, mini-batch gradient descent converges in fewer iterations in comparison to GD LeCun et al. (2012). Another approach to minimise weight fluctuations is parameter tuning. The parameters that directly affect the updates of weights are **learning rate**, **Momentum** and **Weight Decay** which will be discussed in the section below.

4.3.2 Learning by Back Propagation

Weights are initialized and forwarded along with the hidden units during learning. However, initial values of the weights can have a significant effect on the training process as they are to be chosen in such a way that the activation function is primarily activated in its linear region. Backpropagation is the most popular algorithm in training of neural network with supervised learning approach. The basic idea of this algorithm is to apply chain rule to compute the influence of each weight in the network with respect to the cost function $\boldsymbol{E}$. Let $\boldsymbol{w_{ij}}$ be the weight from neuron $\boldsymbol{j}$ to neuron $\boldsymbol{i}$, $\boldsymbol{Y_i}$ be the output and $\boldsymbol{net_i}$ be the weighted sum of the inputs of neuron $\boldsymbol{i}$. The backpropagation algorithm can then be expresses as the chain rule as in Riedmiller and Braun (1993) as:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial Y_i}\frac{\partial Y_i}{\partial net_i}\frac{\partial net_i}{\partial w_{ij}} \tag{4.7}$$

Momentum $\boldsymbol{\alpha}$ is used to minimize the fluctuation in weight changes over consecutive iterations $\boldsymbol{t}$ by supporting the step size to move along the direction of lower curvature region of cost function. Therefore equation(4.2) can be modified by adding the support of momentum as in Riedmiller and Braun (1993) as :

$$\Delta w_{ij}(t) = w_{ij} - \eta \frac{\partial E}{\partial w_{ij}}(t) + \alpha \Delta w_{ij}(t-1) \tag{4.8}$$

Weight decay $\boldsymbol{\beta}$ penalizes the weight changes in each learning steps. Hence considering the weight decay, the weight update can be expresses as in Hanson and Pratt (1989) as:

$$\Delta w_{ij}(t) = w_{ij} - \eta \frac{\partial E}{\partial w_{ij}}(t) - \beta w_{ij}(t-1) \tag{4.9}$$

During back propagation,according to their behaviour, update of weights can be explained in terms of learning rate, momentum and weight decay as:

$$\Delta w_{ij}(t) = w_{ij} - \eta \frac{\partial E}{\partial w_{ij}}(t) + \alpha \Delta w_{ij}(t-1) - \beta w_{ij}(t-1) \tag{4.10}$$

Adaptive learning rate $\boldsymbol{\eta}$: The implementation of adaptive learning rate can bring a stability during weight updates and convergence. It can be done by using a constant $\boldsymbol{a}$ that deceases the rate over a period of time $\boldsymbol{t}$ or epoch as in Xu (2011) as:

$$\eta_{t+1} := \eta_t (\frac{1}{1 + a\eta_t t}) \tag{4.11}$$

4.4 Testing

4.4.1 Overfitting and Underfiting Nature of the Network

To analyse a model that is learning well or not, one can observe the nature of the plot during training and validation based on accuracy and the average loss. Loss is the average of the losses that is computed by the softmax classifier on finding the variation between the actual target value and the output value predicted by the model. Observing the plot, one can get the idea of the capacity of the model. Setting the right number of layers and their size are other important requirements of the network. Increasing the number of layers and the size of the model also increases the capacity of the network. Network with more neurons can express more complicated features and can classify more complicated data. However, the model can also easily suffer from the over-fitting problem. This is also referred to as a high variance problem. In contrast to this problem, it can also easily suffer from the under-fitting problem if the size of the network is not big enough to handle complicated features during training. It is also referred to as a high bias problem. During validation and testing, we can find three general natures of the plots: a high bias, balanced or a high variance which gives an idea of the behaviour of the performance of the network Ng (2015).

High bias is the case of the underfitting model which simply means the model is not deep enough to learn all the necessary features. Therefore, a higher number of layers are desired in these cases to increase the capacity of the model. The performance of such a model is usually poor.

A balance is a case when the accuracy plot is smoothly increasing or the loss plot is smoothly decreasing with the number of the epoch. It is the ideal condition for a good model.

High variance is also referred to as the case of overfitting. That means the model has a higher capacity than required and can easily pick up extra noise with the necessary features. Even in this case, the performance of the model is usually poor.

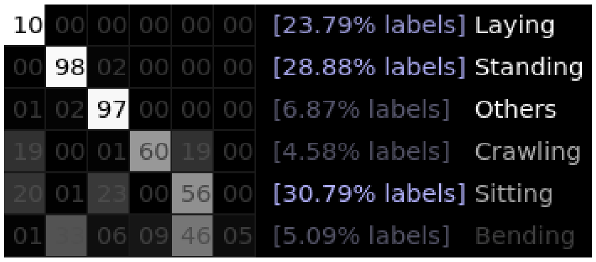
Most cases are over-fitting cases in neural networks. We can reduce overfitting by controlling the number of iteration so that the deeper network does not over-train. We need to stop early sometimes observing the nature of the plots. Another solution to over-fitting is to add more data to the network Nielsen (2016). Regularisation is an important technique that can reduce the problem of overfitting of the neural networks.

Most popular and widely used regularisation is the L2 regularisation where weight is decayed linearly: W += - $\boldsymbol{\lambda}$ * W towards zero. Apart from them, 'Dropout' can also help in regularisation to some extent Li and Karpathy (2015). Dropout is actually used to delete half of the hidden units randomly in between input and output neurons for a temporary period. Now the model will still have to operate its general processes that are forward and backward propagations over mini-batch of examples. This also means that adjusting the weights and the biases will now be with just half of the capacity of hidden units. In the next mini-batch examples, those temporary deleted hidden units are restored and another randomly selected hidden units are deleted and the same process is carried out for updating weights and biases over the mini-batch examples. Therefore, with dropout, weights are compensated using lesser hidden units Nielsen (2016).

4.5 Performance Evaluation Metrics

4.5.1 Confusion Matrix

Confusion matrix can help us to understand the performance of the model on the test. They are used to visualise how well the model is able to classify all the types of classes correctly. It contains information about actual and predicted classifications done by a classification system. The classification error occurs if two different poses are too similar to each other. This is mainly during the transition periods where some poses can be considered as another due to the similarity in the appearance of the pose. Hence, while creating the ground truth, it is very important to look for consistency in labelling of poses as these can easily confuse the model and affect the accuracy. For example, in this study, standing, bending, sitting, crawling and lying poses are considered. Bending and crawling are usually the transitional poses which can be difficult to classify as these can sometimes be treated as another similar pose.



10	00	00	00	00	00	[23.79% labels]	Laying
00	98	02	00	00	00	[28.88% labels]	Standing
01	02	97	00	00	00	[6.87% labels]	Others
19	00	01	60	19	00	[4.58% labels]	Crawling
20	01	23	00	56	00	[30.79% labels]	Sitting
01	33	06	09	46	05	[5.09% labels]	Bending

Figure 4.3: *Example of a confusion matrix*

.

Figure4.3 illustrates a confusion matrix for six different classes according to this classification system drawn during the test. This gives information about the recognition of the classes with respect to the ground truth. In other word, it contains information about actual and predicted classifications done by our classification model on the test images. The diagonal elements represent the number of correctly classified classes and the off-diagonal elements are the number of misclassified classes out of a total number of test images. There are four important decision factors that can be achieved from the confusion matrix to understand the performance of the classification model as explained by Labatut and Cherifi (2012):

- True Positive (TP): Is the proportion of positive cases that were correctly predicted.
- False Positive (FP): Is the proportion of negative cases that were incorrectly predicted as positive.
- True Negative (TN): Is the proportion of negative cases that were correctly predicted as negative.
- False Negative (FN): Is the proportion of positive cases that were incorrectly predicted as negative.

From these cases, one can calculate accuracy and sensitivity which are considered to be the quality criteria for a recognition system. Accuracy is the overall performance of the model and can be calculated as:

$$Accuracy(AC) = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.12}$$

where as sensitivity is the accuracy of the model to predict each class and can be calculated as in Noury et al. (2007) :

$$Sensitivity(SN) = \frac{TP}{TP + FN} \tag{4.13}$$

4.6 Experimental Results and Discussion

The total dataset of 21499 images is divided into three categories: first 15800 images for training, next 3199 images for validation and finally 2500 images for testing.

The proposed model has the following settings as shown below in the table 4.1:

Learning rate	0.015
Weight decay	0.85
minimum learning rate	0.0001
momentum	0.95
maximum epoch	30
batch size	20
training set	15800
validation set	3199
test set	2500

Table 4.1: *Setting details.*

The validation set of data contain those images in which the model has seen the person in the training data. However, these images are from different rooms in comparison to training and test data. There are two different sets of images with a single participant. The two sets are recorded in two different rooms. One of the room is completely unseen in training or in the testing dataset and the other is a room from the test dataset. However, the position of furniture, position of mattresses and view angle have been changed to create a different room out of the room used in the test set. The validation set is very important in the model selection which helps to tune the parameter of the model so that the model is able to provide a meaningful estimation in the test set. Similarly, the test data set contain images in which there are three participants and are never seen in the training dataset or validation dataset. Furthermore, the position of furniture, mattresses and the view angle were also changed to create a different room with all three different participants to achieve a completely unseen dataset for testing.

4.6.1 Accuracy Plot during Validation

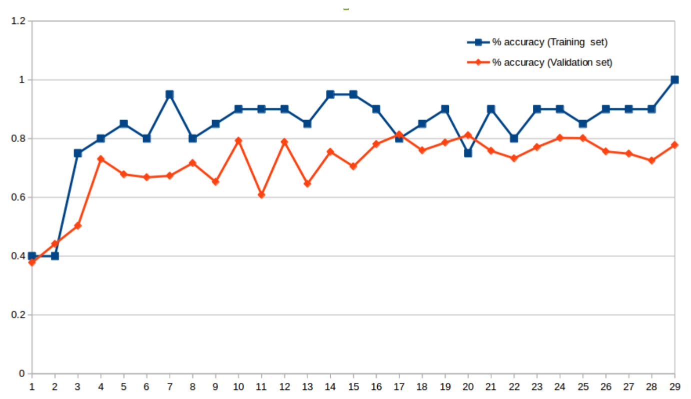


Figure 4.4: *Accuracy plot during validation.*

Figure 4.4 illustrates the behaviour of the CNN model during training on the training dataset and testing on a validation set for accuracy. The blue line represents the nature of accuracy plot on the training set and the red line represents the accuracy plot on the validation set. The plot demonstrates an increase in accuracy as the number of epoch increases. This is a positive sign that the model is learning and improving with every iteration. The training can be stopped once the accuracy reaches to 100% on the training set after which the model is considered to overfit. After performing training on each mini-batch, at the end of the epoch, the model is tested on the validation set and is saved. Moreover, at the end of the second epoch, if the new model after testing is better than the previous model, the new model is saved. Otherwise, the old model is preserved and the new is discarded. In this way, the best model at the end of validation is saved. Therefore, it is not necessary to stop at 100% accuracy during training. The best model is then ultimately tested with test data to get the overall generalised accuracy of the model considering the test data are completely unseen. According to the figure 4.4 shown above, the training set reaches to 100% on 29th iteration. 81% is the best accuracy achieved by testing on validation dataset on the 17th epoch.

4.6.2 Average Loss during Validation

.

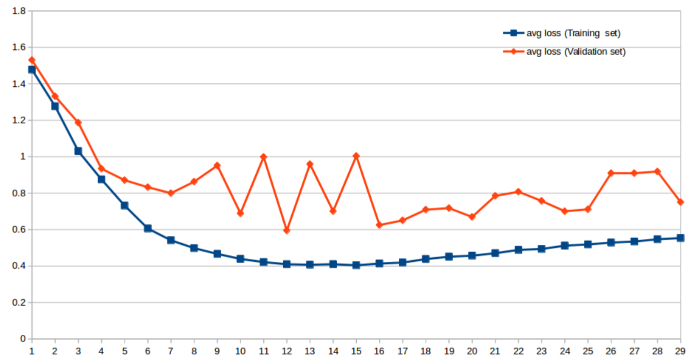
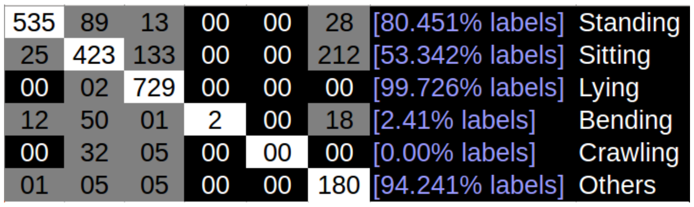


Figure 4.5: *Average loss plot during validation.*

Figure4.5 illustrates the average loss during training on the training dataset and testing on the validation set. The average loss in the training set is represented by the blue line and the loss on the validation dataset is represented by the red line. From the figure 4.5 seen above, the loss has a much smoother curve and is decreasing steadily in the training dataset. However, in the validation test, the loss is not decreasing smoothly and is higher compared to the training set. Comparing with the loss plot of training data, the minimum average error loss on the validation set is 0.6 on the 16th epoch.

4.6.3 Output with Confusion Matrix

Finally, the best model was tested on the test set of images. The overall accuracy on the test set was 74% as illustrated by the output of the test confusion matrix in the figure 4.6 below.



535	89	13	00	00	28	[80.451% labels]	Standing
25	423	133	00	00	212	[53.342% labels]	Sitting
00	02	729	00	00	00	[99.726% labels]	Lying
12	50	01	2	00	18	[2.41% labels]	Bending
00	32	05	00	00	00	[0.00% labels]	Crawling
01	05	05	00	00	180	[94.241% labels]	Others

Figure 4.6: *Output of confusion matrix on test.*

Global correct or overall accuracy is determined by the ratio of the sum of elements in the diagonal/sum of all the elements in the matrix (i.e number of test set). Therefore, Overall Accuracy = 74%.

Similarly, the percentage on the right to the matrix elements associated with the class is the sensitivity of each class. This parameter explains how well the model performed in predicting correctly for each class. It can be computed as: The sensitivity of class 'Lying' = TP/(TP+FN) = 729/(729+(2)) = 99%. Similarly, the sensitivity of class 'standing' is 80%.

From the above confusion matrix, the sensitivity of 'bending' class is 2% and crawling is 0%. This is very poor as the class are not well balanced. More data is required with similar poses for a model to improve. Moreover, the model is confused between standing and bending poses as well as between lying and crawling poses. This is because these bending and crawling are transitional poses and can easily be wrongly classified. Similarly, sitting with 53% is also weak in terms of sensitivity and has also hampered the overall accuracy of the model. However, the model is able to achieve 99% sensitivity in lying pose which is very supportive to the aim of identifying an after-fall pose.

4.7 Pose Recognition on the Unseen Data

Correctly Predicted Scenario

Figure4.7 illustrates four unseen output images: subtracted RGB in the left and subtracted Depth in the right.

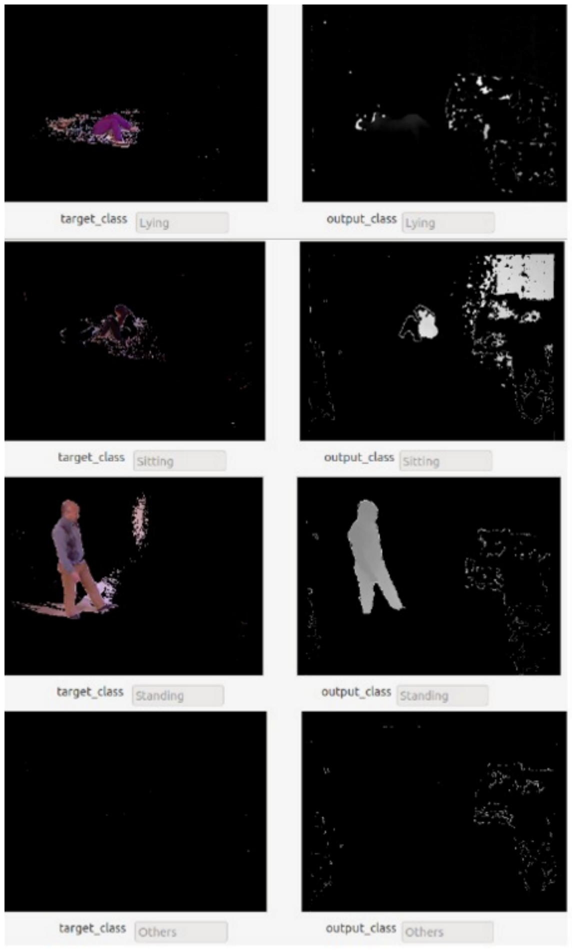


Figure 4.7: *Correctly predicted pose on unseen image.*

The model was able to predict the correct pose with the accuracy of 74%. Most of the correctly predicted pose is clearly visible poses and with less transitional variations. That means the model performs better on clearly visible poses of standing, sitting or lying. Bending and crawling are not predicted well because they are sub-activities that appear similar to other poses and the model can get confused and classify them incorrectly. These complex poses are transitional poses which are observed during the change from one pose to another: standing to sitting or lying and vice-versa.

Incorrectly Predicted Scenario

Figure4.8 illustrates three cases of incorrectly predicted four outputs which are about 26% in the total test set.

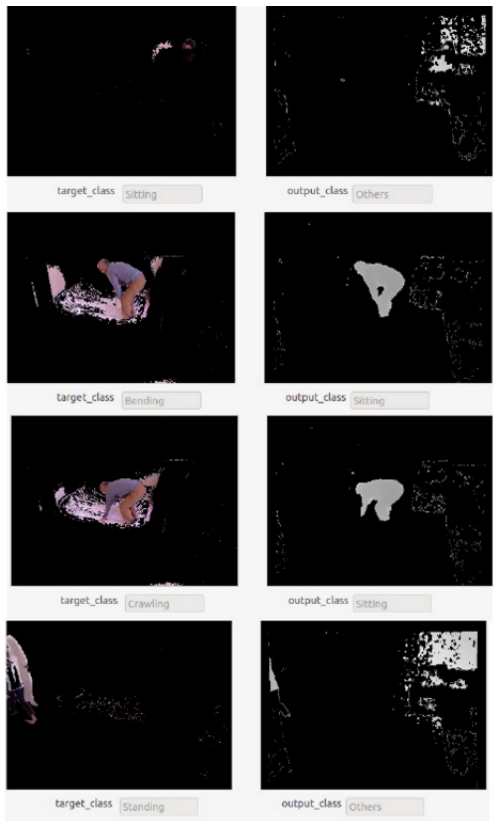


Figure 4.8: *Incorrectly predicted pose on the unseen image.*

The proposed model fails when the person is not seen on the training images in both the input images like the one shown in the first image block above. In actual, seeing the RGB image, the person is sitting on the sofa and is therefore labelled as sitting. However, due to background subtraction and camouflage, the person seems to be missing visually. The model is trained to learn an empty image labelled as ’other’ in training examples. Therefore, the model is predicting as ’other’ in the output class considering the person is absent in the image. A noisy image is also vulnerable and can be wrongly recognised by this model.

In the second case, the person seems to be in transition from bending to standing vice-versa. The prediction goes wrong due to the confusion of the pose. In this case, it is actually labelled as ’bending’. But the model gets confused and can predict either of the one class depending on the number of similar images present in training data. A lesser sensitivity in standing pose is also due to the confusion between standing and bending pose.

It is again the same with the case of crawling in the third case. Hence, the proposed model’s performance was affected by these confusing posses. In the fourth case, only a part of the body is seen while going out or in the case of coming into the room. This is hard to recognise for a model. The model here again does not get enough information about the orientation of the body and therefore does not recognise a pose. It is difficult to even label such partially seen body. However, several images of such types in the training data can improve the prediction.

4.8 Our Contribution

The major contribution of this chapter is that the fusion of RGB and Depth was attempted and used as the input to CNN. RGB and Depth images were recorded from the Kinect sensor for creating our own dataset. Different types of input were tested to the proposed CNN model and the experiment result suggested that combining RGB and Depth after performing background subtraction can be a suitable combination as input as this can help the model to learn in a better way. The result of this experiment can be analysed from the accuracy and loss plots illustrated in the figures below. These are published in our paper Adhikari et al. (2017).

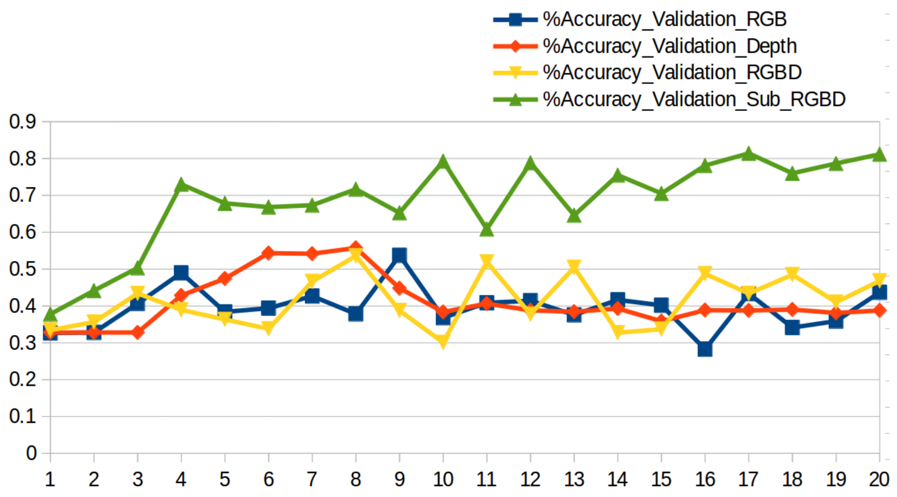


Figure 4.9: *Accuracy plot during test on validation set with different inputs.*

Figure 4.9 illustrates the classification accuracy of the CNN model on a validation set during training with four different types of input images. The plot shows an increase in the accuracy of identifying the correct pose as the number of iteration or epoch increases. This is a positive sign that the model is learning and improving with every iteration. We can stop the training once the accuracy reaches to 100% on the training set after which the model is considered to over-fit. After performing training on each mini-batch, at the end of the epoch, the model is tested on validation set and is saved. Next, at the end of the second epoch, if the new model after testing is better than the previous model, the new model is saved. Otherwise, the old model is preserved and the new is discarded. In this way, we save the best model. Our proposed model was able to achieve the best performance using the subtracted RGBD input with 81% accuracy at the 17th epoch. This can be clearly observed by the green line clearly outperforming all other types.

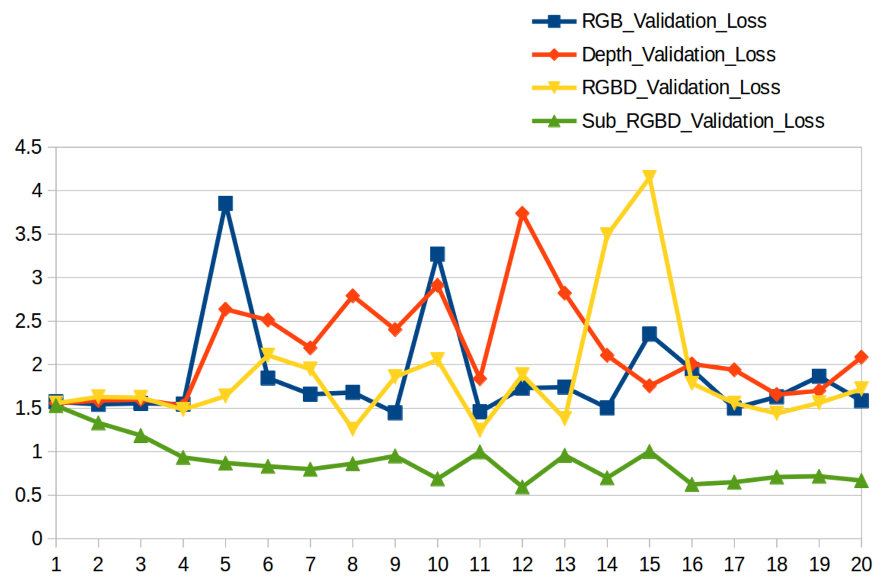


Figure 4.10: *Loss plot during test on validation set with different inputs.*

Similarly, in the fig 4.10, sub-RGBD once again outperforms the rest of the input. Validation loss for sub-RGBD was observed to be the lowest 0.59 at 12th epoch in comparison to other inputs.

4.9 Discussion and Conclusion

In this chapter, a pose recognition model is developed which can use RGB and depth images as input to a convolution neural network for activities classification. Application of RGB and Depth using the convolutional neural network is still not widely explored in the literature for fall detection. The closest work to this study in fall detection for the elderly using RGBD camera was presented by Zhang et al. (2012). They also used the Kinect sensor to record five different activities for recognition which are fall from the chair, fall from standing, standing, sitting on the chair, sit on the floor. They used hierarchy SVM classifier to evaluate the appearance model and the kinematic model for fall detection. They achieved 76% accuracy for the appearance model and 98% accuracy in the kinematic model. They also claim to preserve privacy displaying only depth for monitoring. Another work similar to our in human activity detection using RGBD was presented by Sung et al. (2011). They captured different activities for recognition using RGBD images from the Kinect sensor. They proposed a machine learning method based on hierarchical maximum entropy Markov Model(MEMM) for activity recognition. However, they aimed to train a personal robot to learn the normal activities of the person. They evaluated twelve different activities performed by four different people in five different environments. They were able to achieve the overall performance of 84.3% accuracy in detecting the correct activity where the person is seen before in the training set and 64.2% when the person was not seen before.

In comparison to them, the proposed model was able to achieve 96.27% accuracy during cross-validation test where the person is seen in the training set and 81.29% on the test where the person is never seen in the training. This approach also can preserve privacy displaying only foreground mask from subtracted depth. This system is capable of recognising pose overcoming the problems of viewpoint variation, scaling, deformation, illumination change and partial occlusion. Publicly available datasets were not used at this stage because they were not exactly suitable for this research scenarios since they were created for different other applications. Therefore, specific datasets that contain only five different poses that can be observed during a fall have been created. After this stage, detecting a person for different poses to achieve the location information can be of further advantage. This information can provide a clue to locate a person during different activities including fall. It is also necessary to test on the different publicly available dataset for analysing the performance of the model and achieve generalisation.

Chapter 5

Deep Learning Based Fall Detection Using Simplified Human Posture

5.1 Background

Pose estimation is a challenging task and to add more to this, it is even more challenging when pose estimations are performed on challenging poses that may occur during fall. Location of the body provides a clue where the person is at the time of fall. This paper presents a vision-based tracking strategy where available joints are grouped into three different feature points depending upon the section they are located in the body. The three feature points derived from different joints combinations represents the upper region or head region, mid-region or torso and lower region or leg region. Tracking is always challenging when motion is involved. Hence the idea is to locate the regions in the body in every frame and consider it as the tracking strategy. Grouping these joints can be beneficial to achieve a stable region for tracking. The location of the body parts provides crucial information to distinguish normal activities from falls. A person can be tracked using a computer vision technique during a fall. The pose and location information of the human body part can be used to analyse the occurrence of fall. However, tracking using computer vision on real-time are still an open area of research due to these challenges that are still huge hurdles like lighting variations, the motion of body parts, partial/ fully occlusion and pose deformations. The challenges are even more severe during falls especially on bending or lying pose recognition due to foreshortening distortion. Several important cues for detection are lost during these challenging poses making detection extremely difficult Wang et al. (2012). Following are the questions that demand answers from a fall detection system.

1. Can a person be detected during a fall with rapid variation in poses?
2. Can a person be detected during a fall under different light variations?
3. Can a person be detected under occlusion of any kind?
4. Can the system identify and track human in the presence of another moving object like a pet or a toy?

A system that can defend itself from the above questionnaire is highly desirable in the case of fall detection. This chapter will discuss further regarding our new approach and present its results in the later.

5.2 Our Contribution

This work is mainly inspired by the work of Chua et al. (2015) andSolbach and Tsotsos (2017). The contribution of this study is discussed below:

1. Comparing to the work presented in Chua et al. (2015), the complexity of their approach is further reduced by identifying the person correctly (not moving toys or pets) without the need for background subtraction. The proposed approach will also overcome the major weakness in their approach that they have mention which is to be able to handle different light variations, occlusion and can even be used outdoor to overcome scene and viewpoint variations.
2. Comparing to the work presented in Solbach and Tsotsos (2017), a different low reference point is provided as an alternative to the point on the ground plane. That means there is no need for calculation of ground plane and therefore one can avoid all the complexity in the detection of it and get rid of the weaknesses that were observed when they were missing the ground plane. The detected leg region from the proposed model can be the other lower reference point instead.

5.3 Problem Identification

All the vision-based approach analyses images and videos to identify fall but tend to suffer heavily from the curse of extremely difficult poses that are demonstrated during a fall. Tracking can be accurate during normal activities where the pose is more stable but not during a fall. There are so much of variation in body pose, its speed and also localization. It becomes a big hurdle for a system to achieve better accuracy to fight against such a series of frequently changing poses. Recently pose estimation have reached a new level with joint localization accuracy. Cao et al. (2017) and Güler et al. (2018) are currently the state of the art in pose estimation arena setting a higher benchmark. They are able to achieve a very accurate level of human joint estimation. However, they also suffer a lot while predicting all the joints position mainly when poses are extremely challenging.

The model from Cao et al. (2017) is tested on the challenging images that can be observed during all falls. Their model is capable of identifying 18 joints in the RGB images for multi-person. However, this study aims a single person case and hence only single human cases are examined. The issues identified are shown in figure 5.1. The above images are considered from Adhikari (2017) which the model has never seen before. As illustrated in figure 5.1, the accuracy of all the joints can easily deviate while dealing with difficult poses. Other than that, it is very difficult to predict those joints which are occluded by the body itself.

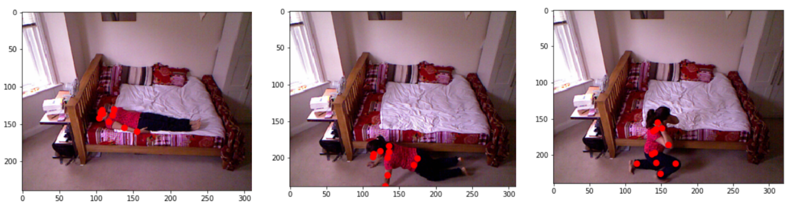


Figure 5.1: *Issues in predicting all the available joints.*

5.4 Our Approach

To solve this problem, this paper looks into a detection based tracking strategy. The idea is to find a stable position of a body part that could be detected all the time during a fall scenario. As shown in the fig 5.2 below, 18 different joints that are available in COCO dataset are grouped into three subgroups to represent three different regions of the body: Upper region or Head region, mid-region or Torso region and lower region or leg region.As shown in the fig 5.2 below, left eye, left ear, right eye, right ear and nose are grouped to represent upper-region or head. Similarly, left shoulder, left elbow, left wrist, right shoulder, right elbow, right wrist and neck are grouped to represent mid-region or Torso. Lastly, left hip, left knee, left feet, right hip, right knee and right feet are grouped to represent lower region or leg. The grouping strategy is proposed with the aim to achieve a stable individual region of a human body. It is not important to identify all the joint positions or body facing side upward or downward to determine fall. To distinguish fall from other normal activities, all that is important is to identify the positions of primary parts of the human body. Simply the height of a head can provide a clue whether a person is standing, sitting or lying. Hence, instead of considering several individual joints, three points were created by grouping individual joints according to the section of the body they could represent.

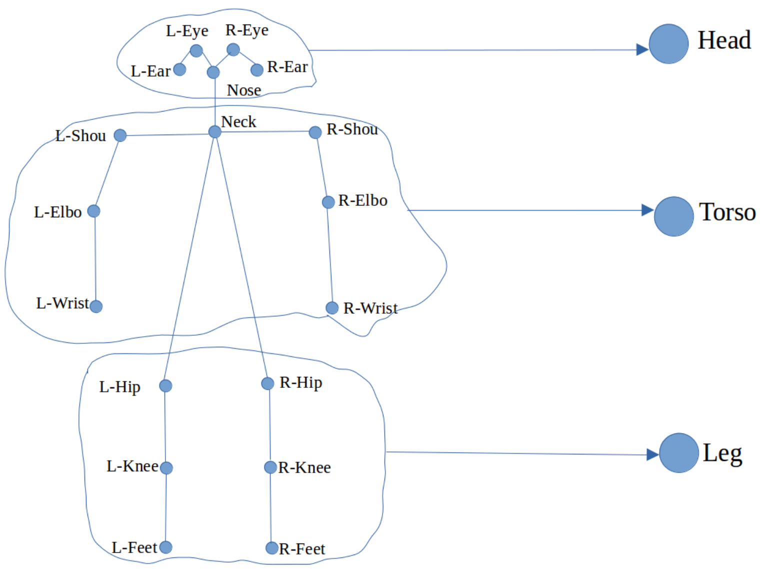


Figure 5.2: *Illustration of joints configuration in COCO dataset grouped into three regions.*

As demonstrated in figure 5.2, the key points are arranged in three individual groups that represent three different regions of the human body. Earlier the model had 18 inputs as there were 18 key points with their confidence value (probability distribution) available in the ground truth. But now these 18 key points are rearranged into three groups. The key points and their corresponding confidence maps for each group are then averaged to achieve one single location and confidence map that represent the center of that group. Therefore, after averaging the key points for each group, there are one central location and confidence map for each group. These are now the three inputs of the model. The model is then also expected to produce three center locations and confidence maps as output. The predicted confidence of the available key points can be averaged to achieve a stable confidence point that can represent that group of nearby joints. Mathematically, the average confidence $\boldsymbol{\Lambda}$ is given by the following equation as in Solbach and Tsotsos (2017):

$$\boldsymbol{\Lambda = \frac{1}{K}\sum_{k=1}^{K}\lambda_k} \tag{5.1}$$

where K is is the number of available detected keypoints and $\boldsymbol{\lambda_k}$ denotes the confidence of the k-th key point.

5.5 Data Augmentation and Training

Since the available joints have been divided into three groups of confidence points, these three points are now fed to CNN model. The model was trained for about a week as COCO datasets 2017 contains 118000 images as training dataset and 5000 as validation dataset. The model was trained using SGD with a batch size of 20, momentum of 0.9 and weight decay of 0.00005. An early stopping strategy was applied to stop the training when the validation loss stops to improve by checking in 5 consecutive epochs. Different data augmentation techniques were implemented to improvise the generalisation of learning. Settings such as rotation range of 30 degrees, a width shift range of 0.2, a height shift range of 0.2, a zoom range of 0.3 and the horizontal flip were used to augment the training data. This technique provides different changes to the original images so that while training the model gets to adopt all the possible changes that can be present in a test image. An adaptive learning rate technique was used to vary the learning rate during the training process. A pre-defined learning rate (LR) of 1e-5 was set. LR was reduced slowly by using step decay to drop the learning rate after certain epochs. The decay factor of 0.1 after every 3 epochs were used during training. The same model which is a modified VGG model and pre-trained weight are used for training as used by Cao et al. (2017). However, a transfer learning technique is further applied to this model as the pre-trained weight already has better learning capacity if trained on similar data. The top layer features are not specific to a particular object and are more general types of features such as edges and colours. Hence, these features can be considered as transferable for different datasets and can be applicable to many other tasks Yosinski et al. (2014). Moreover, the bottom 12 layers are unfrozen to train the model. The higher layers have already learnt the basic features that are necessary for the pose estimation. However, the lower layers that were trained to predict 18 joints are now retrained to predict three sets of points only using the COCO dataset 2017 Lin et al. (2014).

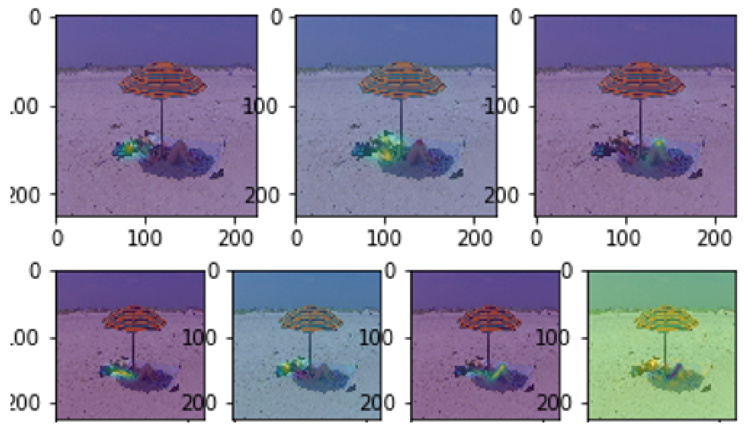


Figure 5.3: *Prediction on validation using COCO dataset.*

Figure 5.3 illustrates the output prediction heatmap and their corresponding PAF on validation using the COCO Dataset. The first row illustrates the average confidence heatmap for the three individual regions centred around that region in three separate images. The second row demonstrates the ground-truth of Part Affinity Field (PAF) connections. Only 2 PAF connections are available in our case: the connection between the upper and middle region is represented by one PAF and the connection between the middle region and the lower region as the second PAF. PAF preserves both the connection and direction information of the body parts. PAF is very efficient in pose estimation to differentiate the actual connection sides like left and right body parts. However, in this case, the direction information from PAF for each connection is unidirectional to locate three centre sub-regions. In the second row, therefore, there are four images demonstrating 2 direction for each PAF respectively.

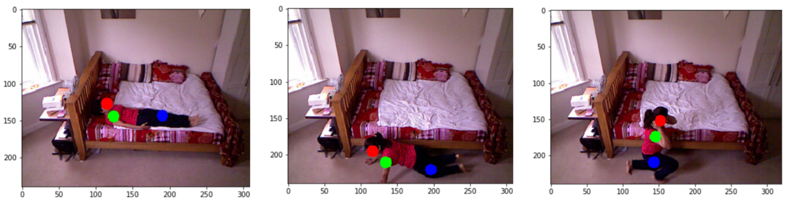


Figure 5.4: *The model output on the same image from fig 5.1.*

Finally, on testing for the same images with the proposed solution to the issues discussed above in figure 5.1, a stable region of detection was achieved successfully. In the above figure 5.4, the three different regions detected are displayed in three colour points with red as the upper region, green as the middle region and blue as the lower region. The best model was saved after validation. The best model was used to detect the three points in a video which was created at a frame rate of 25 frames per second. The model started to predict the three points on the human body within 5 seconds.The rest of the sections looks into different scenario and conditions to analyse the performance of the model.

5.5.1 Challenges:Lighting variations, pose complexity and occlusion

The model performance was tested against five different datasets: COCO datasetLin et al. (2014), Leeds Sports dataset(including extended)Johnson and Everingham (2010), fall dataset from Chua et al. (2015) and Rougier et al. (2013) and our own datasetAdhikari (2017). Both LSP datasets in total contain 12000 different challenging images from the sports arena. Similarly, the other two fall dataset from Chua et al. (2015) and Rougier et al. (2013) contains above 20 videos that are specially created with the aim to analyse an indoor fall for a person. Joints information are not available in these last three fall datasets. Overall these images represent all the challenges a model can easily suffer during a fall. They illustrate variations in poses, scale changes, illumination changes, partial occlusion due to the presence of furniture and also the presence of another object in motion. This may be a presence of a toy, a pet or a stick which could move in parallel with the human. These obstacles can easily confuse the model to identify human as the object of interest in motion.

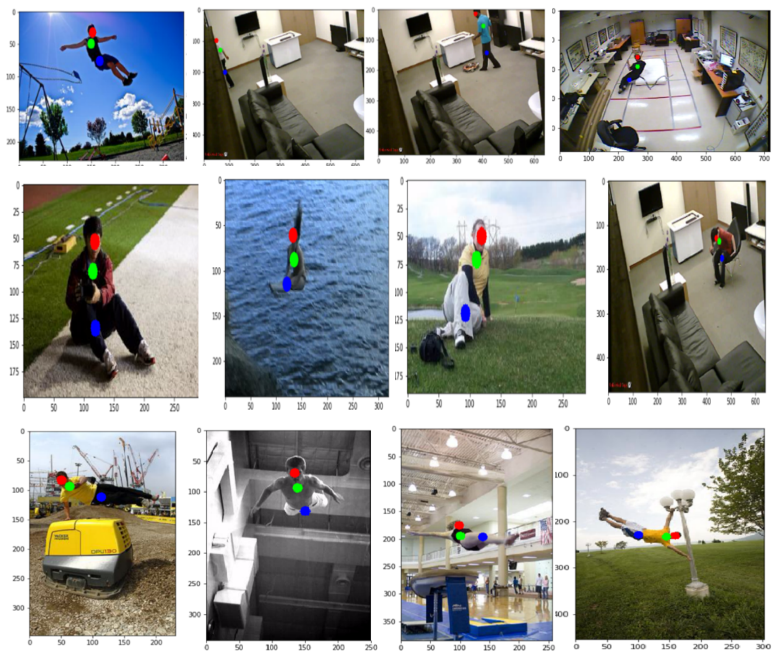


Figure 5.5: *Prediction on LSP, LSP extended and fall dataset from Chua et al. (2015) and Rougier et al. (2013).*

It can be observed from these outputs in Figure 5.5 that our model can detect the postures correctly even during the complex bending, sitting and lying poses in different environments making it more suitable for the real world. From this, one can already get some hint that the lying pose after a fall can be predicted in a more stable way.The research is aimed to detect a fall for a single person or object of interest. However, more than one person may be present in the room. The two persons bodies can overlap at some point. In all the situations, only one detection is possible which is mentioned as a limitation of this research. Multi-person was never considered because one can assist or call for help if another person present in the same room during or after the fall. Other than that, if there is also a moving toy or a pet present, the model is trained to recognize key points on humans only. The model was able to recognize a human and detect the three points who was observed playing with a moving toy demonstrated in the third image from the first row in the above figure 5.5.

5.5.2 Performance Evaluation

To be considered a correct detection, the area of overlap between the predicted bounding box $\boldsymbol{B_p}$ and ground truth bounding box $\boldsymbol{B_{gt}}$ must exceed 50% by the formula as in Everingham et al. (2015):

$$\boldsymbol{a_0 = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})} \geq 50\%} \tag{5.2}$$

Considering the ground truth and the prediction as a centre for three sub-regions, the bounding boxes representing these regions were created of height 10 pixels and width of 10 pixels for each. Fig 5.6 demonstrates the output of our model with the overlapping rectangles of ground truth and output prediction for visual evaluation on LSP dataset. The ground truth bounding boxes of each regions are represented by 'Yellow' bounding box for Head,'Orange' bounding box for Torso and 'Pink' bounding box for lower leg regions.Similarly, for predictions, the bounding boxes that represents the three regions are: 'Red' for Head,'Green' for Torso and 'Blue' for the legs.

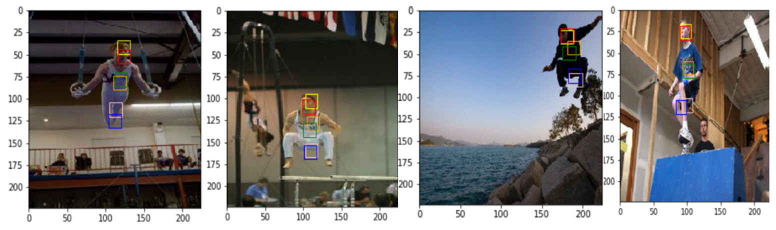


Figure 5.6: *IoU calulation on LSP and LSP extended dataset.*

IoU for each regions is tested on 50 random images from the validation set which contains 1000 images from LSP datasets and the result is presented in the table below:

Dataset	Mean IoU$_{\boldsymbol{Head}}$	Mean IoU$_{\boldsymbol{Torso}}$	Mean IoU$_{\boldsymbol{Leg}}$
LSP+LSP extended	53.84	57.36	50.12

Table 5.1: *Mean IoU for Head, Torso and Leg on LSP dataset.*

The model was tested on 20 videos that contain different activities including fall from the dataset used in Chua et al. (2015). Different number of events were observed from the dataset with respect to the number of events mentioned in their paperChua et al. (2015). It is not mentioned how many videos were tested in total in their paper and therefore it was assumed that only a few numbers of videos were used for their test. Furthermore, different authors have compared posture detection from a different perspective in the literature differentiating each posture in more detail. On the other hand, some of the postures readings are generalised as one posture like forward, backward or side fall to fall only and so as with the case of sitting and bending. The evaluation of our approach on the same dataset provided by Chua et al. (2015) is presented in table 5.2 below with our own perspective.

Events	No.of Events	Correctly Detected	Incorrectly Detected
Fall	30	30	1
Sit	10	10	0
Walk	83	83	0
Run	6	6	0
Squat	5	5	0
kneel	6	5	1

Table 5.2: *Detection of three regions during different activities.*

Demonstrating 18 joints prediction using the model from Cao et al. (2017)and three region based points prediction from the proposed model on LSP extended dataset and video1 from Chua et al. (2015) is shown below in figure 5.7 .

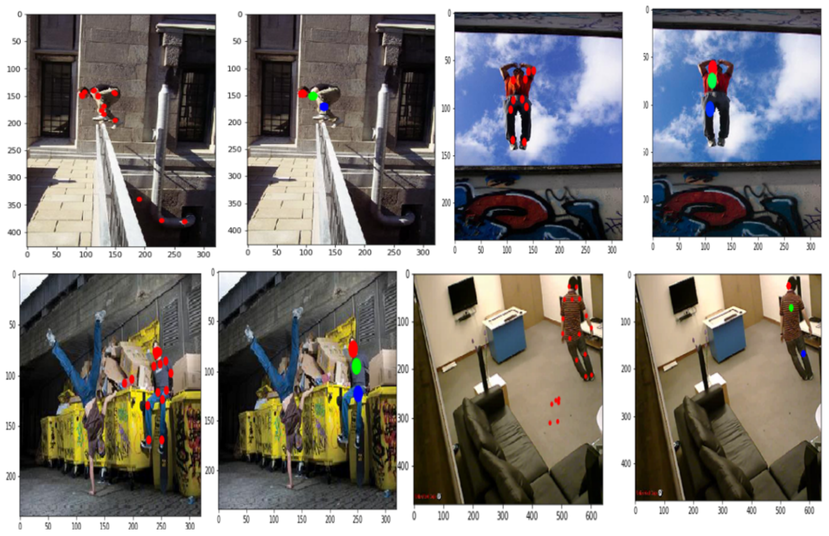


Figure 5.7: *Comparing prediction of 18 joints vs 3 points detection in 4 pair of images.*

5.6 Failure cases

The model although has performed very well to detect the three regions during different scenarios, it has also failed in some cases. As seen in fig 5.8, the first two images represent the problem of deformation of the body that can easily bring complexity in pose estimation. In this case, the body regions were not completed distinguished and the regions seem to overlap.

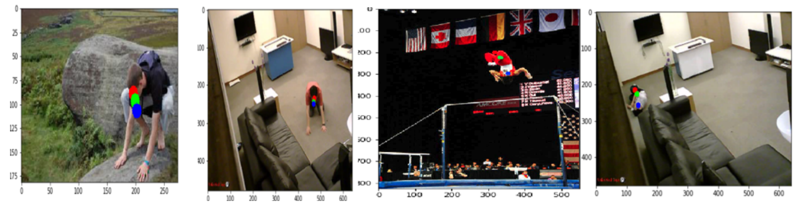


Figure 5.8: . *Failure cases.*

In the latter two images, the person is facing upside down. That means the direction of the head facing towards the bottom side of the image. Similarly, even in the other three images, the person is lying again with the direction of the head towards the bottom of the image.

During a lying pose, y-information is almost same or negligible among the sub-regions. That means only the x-information contributes to learning during a lying pose. It is possible that the training dataset has images where the direction of the human head is mainly towards the top side of the images. Therefore, it can be confusing for the model when the head is towards the downside of the image. This logic can also be justified by another information of the PAF. Usually, PAF is considered for preserving the direction information in pose estimation Cao et al. (2017) where the model learns to identify connections with directions(left and right) among the human body joints. In our case, the sub-regions are the average value represented at the centre of their group of joints. Hence this model understands only the connection between these three regions in one direction and that could be the major reason for this kind of failure. Fine-tuning the model with more images having such cases where the human head is present at the lower end of the image can help to obtain better accuracy. Other than that, using a vertical flip during data augmentation can also be beneficial.

5.7 Conclusion

This paper presents a stable region-based detection strategy that can be used as an alternative to the traditional tracking of human body regions during a fall. The detection of a person that has fallen in real-time is the major interest of this study. The detected three regions can provide significant clues to locate the human body part to distinguish fall from other activities. The proposed model is able to detect region on different postures during fall with 96.7 % accuracy. In comparison to Chua et al. (2015), the detection accuracy was only 90.5 % on the same data due to difficulty in identifying 3 valid regions. Traditional head tracking approach proposed in Rougier et al. (2013) failed in a number of cases due to loss of balance during fall and partial occlusion of the head. Moreover, the model is also able to detect complex pose like kneeling with 83.3 % accuracy. Experimental results indicate that the proposed method can detect and track people during a fall with better accuracy and stability in real-time. This part of the work is available at Adhikari et al. (2019).The possibility of using these region locations as features for classification of fall and non-fall activities is further explored in the next chapter.

Chapter 6

LSTM Based Fall Recognition Using Posture Information

6.1 Background

Fall is a series of change of poses and these changes take place at a higher speed when compared with other normal activities. Therefore to understand this characteristic of fall, one needs to analyse a sequence of frames that contains all kind of activities. However, a fall sequence will demonstrate a faster change of pose that is from standing or sitting to lying and hold a longer lying pose after for a certain period. This is a distinguishing unique characteristic of fall from other normal activities. Pose estimation can be beneficial to analyse the rate of change of pose. It is possible to detect the position of the person with information on body regions or joint positions. The actual posture and position of the body during the activities can be used as the features for classification of action in a sequence of images. The sequence of images can be represented as one whole action and then can be used as ground truth to train a classifier that can handle a whole sequence, learn and make use of the contextual information from the sequence to predict back correctly on similar actions. The basic idea is to make use of temporal information that is the correlations between the images. In general, a sequence of images taken at different time-steps can be used to monitor the dynamic changes observed in the pose of the human body. This will, in turn, provide a significant clue to distinguish fall from other normal activities.

Long Short-Term Memory (LSTM) networks are a type of recurrent neural network capable of learning order dependence from a sequential input. They are able to use contextual information from the input chain using memory cellsSak et al. (2014). Finally, the posture information is feed to LSTM for fall detection. The LSTM model takes two inputs: posture information and the corresponding label for a sequence length to represent an action. Classification of activities is considered in terms of three categories: Class1 as Empty which represent the absence of a person in the scene, class2 as no fall which represent all the normal activities such as standing or sitting and class3 as fall. The fall class sequence is carefully labelled from the start of fall that is when the body starts to tilt or bend towards the ground until the body touches the floor in a stable way. Besides that, all the after fall poses which is mainly lying are considered as no fall.

6.2 Our contribution

The contribution of this chapter is to use the location of the three detected human body region as primary features for the classification of fall. The location information when considered in a sequential data can provide a unique pattern and this pattern can be identified as a clue by the LSTM based recurrent network to differentiate fall from other activities.

6.3 Proposed LSTM Architecture

The model of our LSTM architecture is shown in figure 6.1.

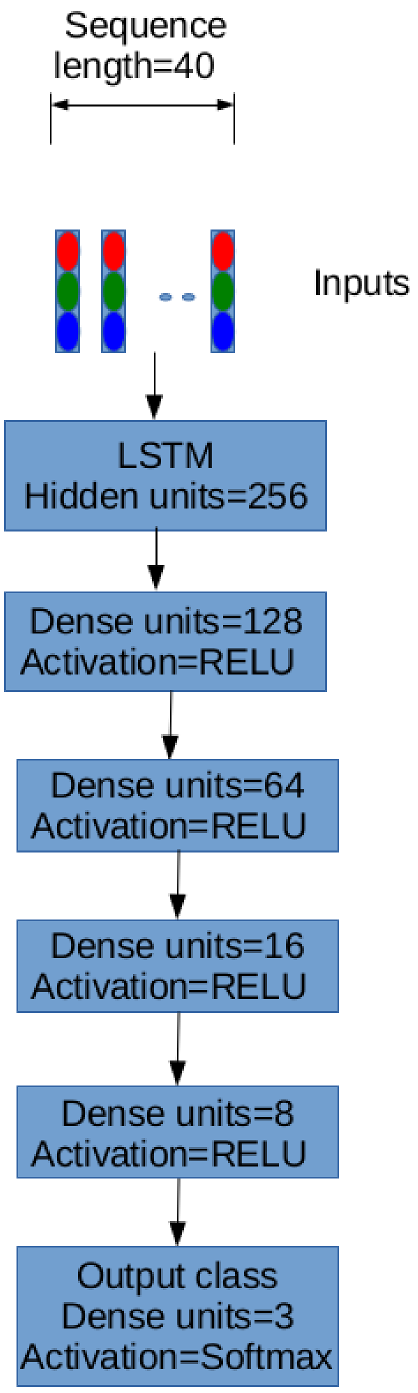

Sequence length=40
Inputs
LSTM Hidden units=256
Dense units=128 Activation=RELU
Dense units=64 Activation=RELU
Dense units=16 Activation=RELU
Dense units=8 Activation=RELU
Output class Dense units=3 Activation=Softmax

Figure 6.1: . *Proposed LSTM architecture.*

In this proposed LSTM architecture in figure 6.1, a fixed length of posture feature sequence is considered as input together with a label that represents the class for the whole sequence. These inputs are fed forward to the LSTM with hidden units of size 256. It is then again followed by another four dense layers with hidden units of size 128, 64,16 and 8. The activation functions in these layers are RELUs. The final dense layer with hidden units of size 3 is the output layer that uses the Softmax activation function for predicting the 3 classes.

The length of the layer and the size of hidden units are solely based on the trial basis as recognised by their impacts on the behaviour of the model.

6.4 Training and validation

The first 22 video datasets were used out of 24 datasets from Rougier et al. (2013) to predict the three regions and then used them as features for classification with the help of LSTM. The datasets are recorded with 8 different views. This is also beneficial for the proposed region based posture detection in case if detection does not work from one view. All the activities are classified into three categories: Empty(absence of person), no fall(activities other than fall) and fall. The number for frame sequences considered is 40 for the classification of each activity as ground-truth. The length is chosen with the aim that a complete action is captured in a sequence.

6.5 Experimental result

The LSTM model is trained with 278 samples and validated on 120 samples with a batch size of 32. Adam is used as the optimizer with the learning rate of 0.0048 for 80 epochs. The overall accuracy of 88.33% on validation is achieved in the 13th epoch as shown in figure 6.2 below. The strategy of saving the best model is implemented whenever the accuracy of the model increases to another higher level. The confusion matrix is used to evaluate the performance of the proposed model.

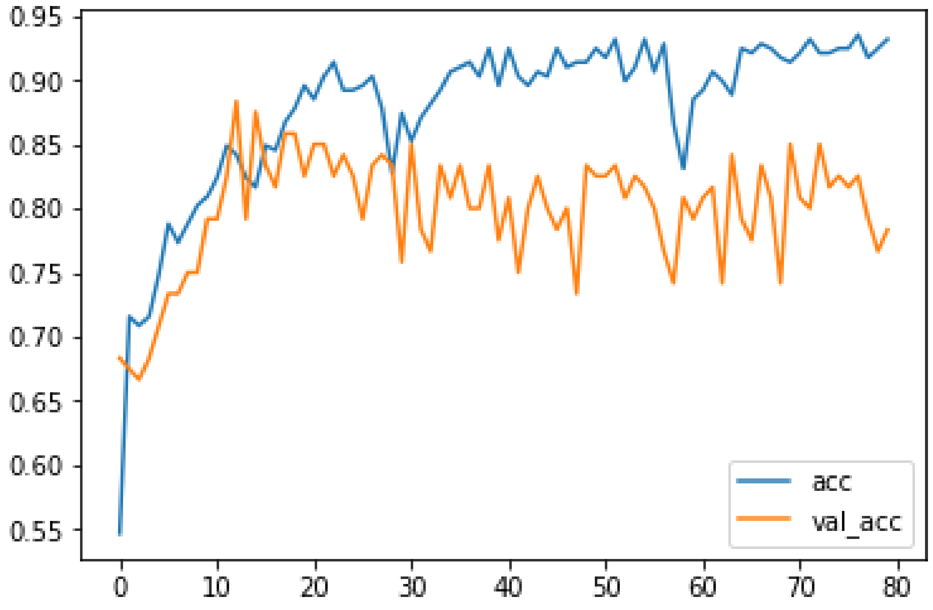


Figure 6.2: . *Training and validation on proposed LSTM.*

Figure 6.3 demonstrates the performance of the classification model in confusion matrix. The number in the diagonal represents the number of correctly predicted samples and the confusion of the model for each class in the rest.

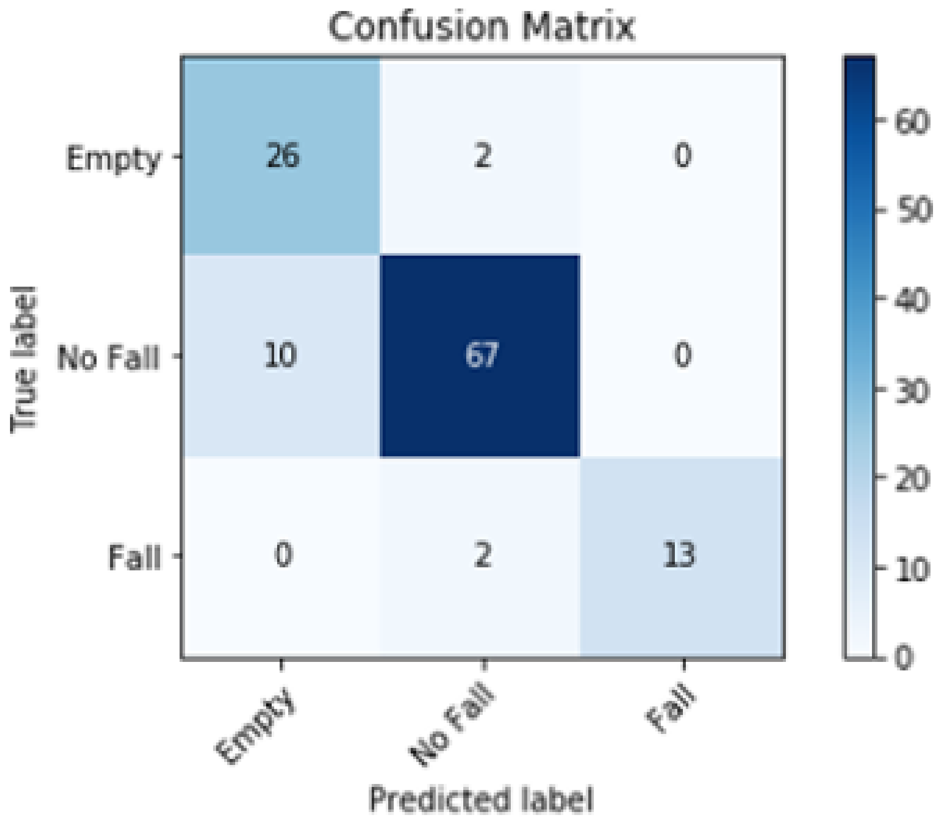


Figure 6.3: . *Performance evaluation using confusion matrix.*

The performance of the proposed model was analysed on the basis of a simple 2 class confusion matrix. Precision, sensitivity or recall and F-score are three important metrics that are widely used in the performance measurements.

The precision metrics inform about how precise or accurate our model is based on the total number of positive predictions. it can be calculated as:

$$Precision = \frac{TruePositive(TP)}{TruePositive(TP) + FalsePositive(FP)} \tag{6.1}$$

The sensitivity or recall is the proportions of falls which are correctly detected. This gives the actual accuracy of the model for detecting only falls correctly. It is calculated as:

$$Recall = \frac{TruePositive(TP)}{TruePositive(TP) + FalseNegative(FN)} \tag{6.2}$$

F-Score is important to balance between precision and recall as accuracy cannot identify the imbalance between the class. A model can have high accuracy despite poor fall detection as it might be good on detecting more non-falls. It can be calculated as:

$$F_{Score} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{6.3}$$

The result from confusion matrix in terms of these metrics for each class are presented in the table 6.1 below:

Class	Precision	Recall	F-Score
Empty	0.72	0.93	0.81
No Fall	0.94	0.87	0.91
Fall	1.00	0.87	0.93

Table 6.1: *Performance metrics for evaluation of fall detection system.*

6.6 Discussion and Conclusion

The precision of class fall is 1.0 which is the most desirable in the case of fall detection. This means that the probability that any alarm raised will be an actual fall and not a false alarm. In the case of the fall class, as seen in the confusion matrix, false positive is zero. Achieving a minimum false positive means the ability of the model to predict correctly is higher. It can be of great advantage especially in the case of fall detection. This means one could avoid the cost that can incur due to the false alarm. Calling the medical team and the ambulance for help is very costly when it is in fact not needed. They could be more efficient somewhere else where the fall has actually occurred.

Similarly, a recall value of 0.87 is observed from the above table for the fall class. Recall value could, therefore, help us to select the best model when there is a high cost associated with a false negative. If a fall detected as no fall or empty, there can be life-threatening consequences for the fallen person. This is also verified from the confusion matrix that 13 falls were correctly predicted out of 15 falls. This is a promising result considering the fact that the number of fall classes in the dataset is lower in number in comparison to the other activities. Getting more fall classes for training in the dataset could improve the detection rate.

The F-measure or F-score of 0.93 is observed from the above table for the fall class. This is promising once again. The F-measure provides a good overview of the performance of the model in terms of fall especially as it considers all outcomes: Detected falls (TP), missed falls (FN) and false alarms (FP) except non-falls (TN) Broadley et al. (2018). It is therefore crucial in a sense that it provides the actual accuracy of the model to detect only falls rather than detection of overall classes that contribute to the total accuracy.

The specificity of our fall class is 100% compared to 98.8% achieved by Wang et al. (2016) on the same multi-view camera based dataset. We were able to achieve 88.33% overall accuracy with the help of LSTM using location features detected from our proposed posture detection strategy.

Chapter 7

Conclusion and Future Work

7.1 Thesis Contribution

The first major contribution is to be able to develop a pose recognition system using the convolution neural network for activities classification using both RGB and Depth images. Application of RGB and Depth using the convolutional neural network is still not completely explored according to the literature, especially for fall detection. The closest work to ours in fall detection for the elderly using RGBD camera was presented by Zhang et al. (2012). They also used Kinect sensor to record 5 different activities for recognition which are fall from the chair, fall from standing, standing, sitting on the chair, sit on the floor. They achieved 76% accuracy for the appearance model. Another work similar to our in human activity detection using RGBD was presented by Sung et al. (2011). They captured different activities for recognition using RGBD images from the Kinect sensor. They were able to achieve the overall performance of 84% accuracy in detecting the correct activity where the person is seen before in the training set and 64.2% when the person was not seen before. In comparison to them, we are able to achieve 81%

accuracy during validation test where the person is seen but not the room in the training set and 74% on the test where the person and the room are never seen in the training set. The sensitivity of lying pose is 99% which is extremely desirable in fall detection where an after-fall pose is considered to be lying. Our approach also can preserve privacy displaying only foreground mask from subtracted depth. This system is capable of recognising pose overcoming almost all the challenges that exist in computer vision to some extent.

Secondly, the location of the person can better explain where the person was during the fall. Hence, pose estimation was explored where human joints can be predicted to understand the location of the human. But pose estimation are not stable in a falling sequence. The problem in pose estimation was identified, especially during fall. Fall adds further complexity due to difficult poses that change at a higher speed. Therefore, a solution is proposed which is to group the nearby joints to detect three stable regions of the body. Location of these regions can provide a significant clue for distinguishing fall from other normal activities. The model was able to detect three body regions with 96.7% accuracy in comparison to Chua et al. (2015) who were able to achieve only 90.5% accuracy on the same dataset. The three regions based posture detection was also tested on several publicly available datasets.

Finally, these locations information from the three-body region of the postures were used as features to an LSTM based recurrent neural network. The location information was feed in a sequence of the length of 40 frames to the LSTM to identify the unique pattern that a fall has from other normal activities. The ground truth was created in the form of three classes to demonstrate fall, no-fall and empty cases. The features were labelled as classes and used as input for the proposed LSTM recurrent network to classify fall from other activities. The model was able to achieve 88.33% accuracy with our proposed LSTM model. Comparing the specificity of the fall class, the proposed LSTM model was able to achieve 100% specificity while Wang et al. (2016) were able to achieve only 98.8%.

7.2 Limitations

Following are the limitations of our research work:

- Although at the initial stage, RGB and Depth were both used as input for pose classification, the later stages used only RGB input. Limited availability of depth images for pose estimation and limited distance information were major issues in their deployment in the later stages.
- Posture detection fails when the fall ends up opposite to the camera angle that is head facing towards the bottom of the image. This can affect the classification using LSTM if the posture information is not recorded for a fall in such direction.
- The system is trained with images where only one person is present. The system does not recognise multiple persons in the same frame and therefore there is a risk that a fallen person may not be detected in the presence of another person.
- The other limitation of this research is that it does not run in one attempt. Different stages are combined to finally achieve fall classification.

7.3 Future work

The recommended future work for this research are as follows:

- In future, it is recommended to training the model using more balance data for fall and normal activities to improve the accuracy for posture detection.
- Data augmentation settings can be further explored where the person can be rotated to a greater angle, use vertical and horizontal flips to enhance the learning of the model for detection. Better posture detection can improve the classification accuracy of LSTM in the final stage.
- Training the model with night scenes where the visibility is poor can also be beneficial. This can help in the further generalisation of the model.
- Test the detected region features with other classification techniques to explore further possibilities in fall detection.
- In future, one can try to track a falling person in the presence of another person.

Chapter 8

List of Publications

Adhikari, K., Bouchachia, H. and Nait-Charif, H., 2017, May. Activity recognition for indoor fall detection using convolutional neural network. In 2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA) (pp. 81-84). IEEE.

Adhikari, K., Bouchachia, H. and Nait-Charif, H., 2019.Long Short-Term Memory Networks Based Fall Detection Using Unified Pose Estimation. In 2019 Twelfth International Conference on Machine Vision (ICMV 2019). SPIE.

References

Adhikari, K., 2017. Fall detection dataset. URL:`http://www.falldataset.com/` [Last Accessed 12 October 2018].

Adhikari, K., Bouchachia, H., Nait-Charif, H., 2017. Activity recognition for indoor fall detection using convolutional neural network, in: 2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA), IEEE. pp. 81–84.

Adhikari, K., Bouchachia, H., Nait-Charif, H., 2019. Deep learning based fall detection using simplified human posture. International Journal of Computer and Systems Engineering 13, 255–260.

Age, U., 2016. Later life in the united kingdom. Age UK Factsheet .

Alzahrani, M.S., Jarraya, S.K., Ben-Abdallah, H., Ali, M.S., 2019. Comprehensive evaluation of skeleton features-based fall detection from microsoft kinect v2. Signal, Image and Video Processing , 1–9.

Bahdanau, D., Cho, K., Bengio, Y., 2014. Neural machine translation by jointly learning to align and translate. ArXiv preprint arXiv:1409.0473 .

Bian, Z.P., Hou, J., Chau, L.P., Magnenat-Thalmann, N., 2015. Fall detection based on body part tracking using a depth camera. IEEE journal of biomedical and health informatics 19, 430–439.

Bottou, L., 2010. Large-scale machine learning with stochastic gradient descent, in: Proceedings of COMPSTAT'2010. Springer, pp. 177–186.

Broadley, R., Klenk, J., Thies, S., Kenney, L., Granat, M., 2018. Methods for the real-world evaluation of fall detection technology: a scoping review. Sensors 18, 2060.

Buys, K., Cagniart, C., Baksheev, A., De Laet, T., De Schutter, J., Pantofaru, C., 2014. An adaptable system for rgb-d based human body detection and pose estimation. Journal of visual communication and image representation 25, 39–52.

Cao, Z., Simon, T., Wei, S.E., Sheikh, Y., 2017. Realtime multi-person 2d pose estimation using part affinity fields, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7291–7299.

Cauwenberghs, G., 1993. A fast stochastic error-descent algorithm for supervised learning and optimization. Advances in neural information processing systems , 244–244.

CDC, W.C.H.A., 2015. Important facts about falls .

Chua, J.L., Chang, Y.C., Lim, W.K., 2015. A simple vision-based fall detection technique for indoor video surveillance. Signal, Image and Video Processing 9, 623–633.

Delahoz, Y.S., Labrador, M.A., 2014. Survey on fall detection and fall prevention using wearable and external sensors. Sensors 14, 19806–19842.

Dionyssiotis, Y., 2012. Analyzing the problem of falls among older people. International journal of general medicine 5, 805.

Doulamis, A., Doulamis, N., 2018. Adaptive deep learning for a vision-based fall detection, in: Proceedings of the 11th PErvasive Technologies Related to Assistive Environments Conference, ACM. pp. 558–565.

Dunnell, K., 2008. Ageing and mortality in the uk: National statistician's annual article on the population. Population trends , 6.

Durrant-Whyte, H., Roy, N., Abbeel, P., 2012. Lying pose recognition for elderly fall detection. Robotics:Science and Systems 7, 345–353.

Everingham, M., Eslami, S.M.A., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A., 2015. The pascal visual object classes challenge: A retrospective. International Journal of Computer Vision 111, 98–136.

Ezatzadeh, S., Keyvanpour, M.R., 2019. Vifa: an analytical framework for vision-based fall detection in a surveillance environment. Multimedia Tools and Applications , 1–23.

Fan, K., Wang, P., Zhuang, S., 2019. Human fall detection using slow feature analysis. Multimedia Tools and Applications 78, 9101–9128.

Fu, Z., Culurciello, E., Lichtsteiner, P., Delbruck, T., 2008. Fall detection using an address-event temporal contrast vision sensor, in: Circuits and Systems, 2008. ISCAS 2008. IEEE International Symposium on, IEEE. pp. 424–427.

Gasparrini, S., Cippitelli, E., Spinsante, S., Gambi, E., 2014. A depth-based fall detection system using a kinect® sensor. Sensors 14, 2756–2775.

Güler, R.A., Neverova, N., Kokkinos, I., 2018. Densepose: Dense human pose estimation in the wild. arXiv preprint arXiv:1802.00434 .

Halima, I., Laferté, J.M., Cormier, G., Fougère, A.J., Dillenseger, J.L., 2019. Sensors fusion for head tracking using particle filter in a context of falls detection.

Hanson, S.J., Pratt, L.Y., 1989. Comparing biases for minimal network construction with back-propagation, in: Advances in neural information processing systems, pp. 177–185.

Heinrich, S., Rapp, K., Rissmann, U., Becker, C., König, H.H., 2010. Cost of falls in old age: a systematic review. Osteoporosis international 21, 891–902.

Henry, P., Krainin, M., Herbst, E., Ren, X., Fox, D., 2012. Rgb-d mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments. The International Journal of Robotics Research 31, 647–663.

Hyndman, D., Ashburn, A., Stack, E., 2002. Fall events among people with stroke living in the community: circumstances of falls and characteristics of fallers. Archives of physical medicine and rehabilitation 83, 165–170.

Iazzi, A., Rziza, M., Thami, R.O.H., 2018. Fall detection based on posture analysis and support vector machine, in: Advanced Technologies for Signal and Image Processing (ATSIP), 2018 4th International Conference on, IEEE. pp. 1–6.

Ioffe, S., Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 .

Jain, A., Tompson, J., Andriluka, M., Taylor, G.W., Bregler, C., 2013. Learning human pose estimation features with convolutional networks. arXiv preprint arXiv:1312.7302 .

Jang, D.S., Jang, S.W., Choi, H.I., 2002. 2d human body tracking with structural kalman filter. Pattern Recognition 35, 2041–2049.

Jansen, B., Deklerck, R., 2006. Context aware inactivity recognition for visual fall detection, in: 2006 Pervasive Health Conference and Workshops, IEEE. pp. 1–4.

Ji, S., Xu, W., Yang, M., Yu, K., 2013. 3d convolutional neural networks for human action recognition. Pattern Analysis and Machine Intelligence, IEEE Transactions on 35, 221–231.

Jiang, M., Chen, Y., Zhao, Y., Cai, A., 2013. A real-time fall detection system based on hmm and rvm, in: Visual Communications and Image Processing (VCIP), 2013, IEEE. pp. 1–6.

Johnson, S., Everingham, M., 2010. Clustered pose and nonlinear appearance models for human pose estimation, in: Proceedings of the British Machine Vision Conference. Doi:10.5244/C.24.12.

Kangas, M., Vikman, I., Wiklander, J., Lindgren, P., Nyberg, L., Jämsä, T., 2009. Sensitivity and specificity of fall detection in people aged 40 years and over. Gait & posture 29, 571–574.

Karnin, E.D., 1990. A simple procedure for pruning back-propagation trained neural networks. Neural Networks, IEEE Transactions on 1, 239–242.

Karpathy, A., 2015. The unreasonable effectiveness of recurrent neural networks. Andrej Karpathy blog 21.

Kotsiantis, S.B., Zaharakis, I., Pintelas, P., 2007. Supervised machine learning: A review of classification techniques.

Labatut, V., Cherifi, H., 2012. Accuracy measures for the comparison of classifiers. arXiv preprint arXiv:1207.3790 .

LeCun, Y., Kavukcuoglu, K., Farabet, C., et al., 2010. Convolutional networks and applications in vision., in: ISCAS, pp. 253–256.

LeCun, Y.A., Bottou, L., Orr, G.B., Müller, K.R., 2012. Efficient backprop, in: Neural networks: Tricks of the trade. Springer, pp. 9–48.

Li, F.F., Karpathy, A., 2015. Cs231n: Convolutional neural networks for visual recognition.

Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft coco: Common objects in context, in: European conference on computer vision, Springer. pp. 740–755.

Ludwig, J., 2015. Image convolution .

Luštrek, M., Kaluža, B., 2009. Fall detection and activity recognition with machine learning. Informatica 33.

Ma, X., Wang, H., Xue, B., Zhou, M., Ji, B., Li, Y., 2014. Depth-based human fall detection via shape features and improved extreme learning machine. Biomedical and Health Informatics, IEEE Journal of 18, 1915–1922.

de Miguel, K., Brunete, A., Hernando, M., Gambao, E., 2017. Home camera-based fall detection system for the elderly. Sensors 17, 2864.

Min, W., Zou, S., Li, J., 2019. Human fall detection using normalized shape aspect ratio. Multimedia Tools and Applications 78, 14331–14353.

Mohamed, O., Choi, H.J., Iraqi, Y., 2014. Fall detection systems for elderly care: a survey, in: New Technologies, Mobility and Security (NTMS), 2014 6th International Conference on, IEEE. pp. 1–4.

Mubashir, M., Shao, L., Seed, L., 2013. A survey on fall detection: Principles and approaches. Neurocomputing 100, 144–152.

Nait-Charif, H., McKenna, S.J., 2004. Activity summarisation and fall detection in a supportive home environment, in: Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on, IEEE. pp. 323–326.

Ng, A., 2015. Machine learning video lecture, week 1, gradient descent intuition, stanford university. URL:https://www.coursera.org/learn/machine-learning/lecture/GFFPB/gradient-descent-intuition [Last Accessed 20 April 2016].

Nielsen, 2016. Improving the way neural networks learn. URL:http://neuralnetworksanddeeplearning.com/chap3.html/ [Last Accessed 19 March 2016].

Noury, N., Fleury, A., Rumeau, P., Bourke, A., Laighin, G., Rialle, V., Lundy, J., 2007. Fall detection-principles and methods, in: Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE, IEEE. pp. 1663–1666.

Núñez-Marcos, A., Azkune, G., Arganda-Carreras, I., 2017. Vision-based fall detection with convolutional neural networks. Wireless Communications and Mobile Computing 2017.

Olah, C., 2015. Understanding lstm networks .

OPENCV, 2014. How to use background subtraction methods.

Panahi, L., Ghods, V., 2018. Human fall detection using machine vision techniques on rgb–d images. Biomedical Signal Processing and Control 44, 146–153.

Pascanu, R., Mikolov, T., Bengio, Y., 2012. Understanding the exploding gradient problem. CoRR, abs/1211.5063 2.

Pascanu, R., Mikolov, T., Bengio, Y., 2013. On the difficulty of training recurrent neural networks, in: International conference on machine learning, pp. 1310–1318.

Phi, M., 2019. Illustrated guide to lstms and grus: A step by step explanation. URL:http://www.kurious.pub/blog/Illustrated-Guide-to-LSTMs-and-GRUs-A-step-by-step-explanation-61 [Last Accessed 14 July 2019].

Riedmiller, M., Braun, H., 1993. A direct adaptive method for faster backpropagation learning: The rprop algorithm, in: Neural Networks, 1993., IEEE International Conference on, IEEE. pp. 586–591.

Rougier, C., Auvinet, E., Rousseau, J., Mignotte, M., Meunier, J., 2011. Fall detection from depth map video sequences, in: Toward useful services for elderly and people with disabilities. Springer, pp. 121–128.

Rougier, C., Meunier, J., St-Arnaud, A., Rousseau, J., 2006. Monocular 3d head tracking to detect falls of elderly people, in: Engineering in Medicine and Biology Society, 2006. EMBS'06. 28th Annual International Conference of the IEEE, IEEE. pp. 6384–6387.

Rougier, C., Meunier, J., St-Arnaud, A., Rousseau, J., 2013. 3d head tracking for fall detection using a single calibrated camera. Image and Vision Computing 31, 246–254.

Sak, H., Senior, A., Beaufays, F., 2014. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. arXiv preprint arXiv:1402.1128 .

Shahroudy, A., Liu, J., Ng, T.T., Wang, G., 2016. Ntu rgb+ d: A large scale dataset for 3d human activity analysis, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1010–1019.

Sharif, S.I., Al-Harbi, A.B., Al-Shihabi, A.M., Al-Daour, D.S., Sharif, R.S., 2018. Falls in the elderly: assessment of prevalence and risk factors. Pharmacy practice 16.

Shoaib, M., Dragon, R., Ostermann, J., 2010. View-invariant fall detection for elderly in real home environment, in: Image and Video

Technology (PSIVT), 2010 Fourth Pacific-Rim Symposium on, IEEE. pp. 52–57.

Shojaei-Hashemi, A., Nasiopoulos, P., Little, J.J., Pourazad, M.T., 2018. Video-based human fall detection in smart homes using deep learning, in: Circuits and Systems (ISCAS), 2018 IEEE International Symposium on, IEEE. pp. 1–5.

Shotton, J., Sharp, T., Kipman, A., Fitzgibbon, A., Finocchio, M., Blake, A., Cook, M., Moore, R., 2013. Real-time human pose recognition in parts from single depth images. Communications of the ACM 56, 116–124.

Simonyan, K., Zisserman, A., 2014. Two-stream convolutional networks for action recognition in videos, in: Advances in Neural Information Processing Systems, pp. 568–576.

Skymind, 2016. Deep Learning for Java, convolutional networks. `URL: http://deeplearning4j.org/convolutionalnets.html/` [Last Accessed on:2016-03-15].

Solbach, M.D., Tsotsos, J.K., 2017. Vision-based fallen person detection for the elderly, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 1433–1442.

Soomro, K., Zamir, A.R., Shah, M., 2012. Ucf101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402 .

Stone, E.E., Skubic, M., 2015. Fall detection in homes of older adults using the microsoft kinect. Biomedical and Health Informatics, IEEE Journal of 19, 290–301.

Sung, J., Ponce, C., Selman, B., Saxena, A., 2011. Human activity detection from rgbd images. plan, activity, and intent recognition 64.

Sylliaas, H., Idland, G., Sandvik, L., Forsen, L., Bergland, A., 2009. Does mortality of the aged increase with the number of falls? results from a nine-year follow-up study. European journal of epidemiology 24, 351–355.

Talathi, S.S., Vartak, A., 2015. Improving performance of recurrent neural network with relu nonlinearity. arXiv preprint arXiv:1511.03771 .

Toshev, A., Szegedy, C., 2014. Deeppose: Human pose estimation via deep neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1653–1660.

Vaidehi, V., Ganapathy, K., Mohan, K., Aldrin, A., Nirmal, K., 2011. Video based automatic fall detection in indoor environment, in: Recent Trends in Information Technology (ICRTIT), 2011 International Conference on, IEEE. pp. 1016–1020.

Wang, S., Chen, L., Zhou, Z., Sun, X., Dong, J., 2016. Human fall detection in surveillance video based on pcanet. Multimedia tools and applications 75, 11603–11613.

Wang, S., Zabir, S., Leibe, B., 2012. Lying pose recognition for elderly fall detection. Robotics: Science and Systems VII 345.

WHO, 2007. Who global report on falls prevention in older age. `URL:http://www.who.int/violence_injury_prevention/publications/other_injury/falls_prevention.pdf?ua=1` [Last Accessed 3 December 2016].

Wu, G., 2000. Distinguishing fall activities from normal activities by velocity characteristics. Journal of biomechanics 33, 1497–1500.

Wu, R., Yan, S., Shan, Y., Dang, Q., Sun, G., 2015. Deep image: Scaling up image recognition. arXiv preprint arXiv:1501.02876 22, 388.

Xu, T., Zhou, Y., Zhu, J., 2018. New advances and challenges of fall detection systems: A survey. Applied Sciences 8, 418.

Xu, W., 2011. Towards optimal one pass large scale learning with averaged stochastic gradient descent. arXiv preprint arXiv:1107.2490 .

Yeung, S., Russakovsky, O., Jin, N., Andriluka, M., Mori, G., Fei-Fei, L., 2018. Every moment counts: Dense detailed labeling of actions in complex videos. International Journal of Computer Vision 126, 375–389.

Yosinski, J., Clune, J., Bengio, Y., Lipson, H., 2014. How transferable are features in deep neural networks?, in: Advances in neural information processing systems, pp. 3320–3328.

You, Q., Jin, H., Wang, Z., Fang, C., Luo, J., 2016. Image captioning with semantic attention, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4651–4659.

Yu, M., Naqvi, S.M., Chambers, J., 2009. Fall detection in the elderly by head tracking, in: Statistical Signal Processing, 2009. SSP'09. IEEE/SP 15th Workshop on, IEEE. pp. 357–360.

Yu, X., 2008. Approaches and principles of fall detection for elderly and patient, in: e-health Networking, Applications and Services, 2008. HealthCom 2008. 10th International Conference on, IEEE. pp. 42–47.

Zaremba, W., Sutskever, I., Vinyals, O., 2014. Recurrent neural network regularization. arXiv preprint arXiv:1409.2329 .

Zhang, C., Tian, Y., Capezuti, E., 2012. Privacy preserving automatic fall detection for elderly using RGBD cameras. Springer.

Zhang, M., Zhang, Y., Vo, D.T., 2016. Gated neural networks for targeted sentiment analysis, in: Thirtieth AAAI Conference on Artificial Intelligence.

Zhang, Z., Conly, C., Athitsos, V., 2015. A survey on vision-based fall detection, in: Proceedings of the 8th ACM international conference on PErvasive technologies related to assistive environments, ACM. p. 46.