Vision based Human Activity Recognition using Skeleton Data

Author

Sumaira Ghazal

00000118185

Supervisor

Dr. Umar Shahbaz Khan

DEPARTMENT OF MECHATRONICS ENGINEERING

COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY

ISLAMABAD

January, 2019

Vision based Human Activity Recognition using Skeleton Data

Author

Sumaira Ghazal

00000118185

Submitted to the Department of Mechatronics Engineering in partial fulfillment of the requirements for the degree of

MS Mechatronics Engineering

Thesis Supervisor:

Dr. Umar Shahbaz Khan

Supervisor's Signature: _____

DEPARTMENT OF MECHATRONICS ENGINEERING

COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY

ISLAMABAD

January, 2019

Declaration

I certify that I have developed this thesis titled "*Vision based Human Activity Recognition using Skeleton Data"* with my own personal efforts and under the guidance of my supervisor Dr Umar Shahbaz Khan. The work has not been presented elsewhere for assessment. All the material used from other sources has been properly cited.

Sumaira Ghazal

00000118185

Language Correctness Certificate

This thesis has been read by an English expert and is free of typing, syntax, semantic, grammatical and spelling mistakes. Thesis is also according to the format given by the university.

Sumaira Ghazal

00000118185

Dr. Umar Shahbaz Khan

(Supervisor)

Copyright Statement

Acknowledgements

First of all, I would like to thank Allah Almighty who gave me the ability and bestowed me with perseverance to complete this thesis. His works are truly splendid and wholesome and His knowledge is truly complete with due perfection.

I am truly thankful to my supervisor Dr. Umar Shahbaz Khan for his expert advice and guidance throughout this thesis.

I would like to acknowledge and thank my thesis guidance and evaluation committee members, Dr. Waqar Shahid Qureshi and Dr. Mohsin Islam Tiwana. I am also thankful to all my degree-fellows and friends for their help and cooperation.

I would also like to thank the authors of the Openpose Library, Gines Hidalgo, Zhe Cao, Tomas Simon, Shih-En Wei, Hanbyul Joo, and Yaser Sheikh, whose work laid the foundation for this thesis.

I am gratefully thankful for the support and encouragement of my parents. I would also like to appreciate and thank my husband for his continuous motivation and support throughout my thesis work.

Dedicated to my beloved family for their endless love, support and encouragement

Abstract

Vision based Human Activity Recognition or simply HAR is a widely researched area that is helpful in understanding human behaviour in images and videos. HAR is an important part of various research problems such as detecting and preventing crimes with the help of automated video surveillance, robot movement without human intervention and to provide telecare for elderly. In this research, an algorithm for activity recognition using 2D pose information extracted from human skeleton is implemented. The approach is based on angles between the joints and displacement of joints between frames. Two publically available datasets are used for training and testing purpose. For activity recognition, five well known techniques of supervised machine learning are implemented separately including K nearest neighbours, SVM, Linear Discriminant, Naïve Bayes and Back propagation neural network. Using these techniques, four action classes Sit, Stand, Fall and Walk, are recognized in videos. Results for all the classifiers are compared to find the best performing technique for the proposed methodology. All classifiers performed well with the best performing classifier achieving an overall accuracy of 98%. The results show that proposed methodology gives compatible accuracy with the state of the art in this field.

Key Words: *Human activity recognition, supervised machine learning, skeleton features*

Table of Contents

Declaration ..... **i**
Language Correctness Certificate ..... **ii**
Copyright Statement ..... **iii**
Acknowledgements ..... **iv**
Abstract ..... **vi**
Table of Contents ..... **vii**
List of Figures ..... **ix**
List of Tables ..... **x**
CHAPTER 1: INTRODUCTION **1**
1.1 Motivation, Scope and Background 1
1.1.1 Categories of HAR 1
1.1.2 Applications of HAR 2
1.2 Research Objectives 3
1.3 Thesis Organization 4
CHAPTER 2: LITERATURE REVIEW **5**
2.1 Methods using skeletal features 5
2.2 Methods using silhouette based features 6
2.3 Methods using motion features 7
2.4 Methods using template matching techniques 9
2.5 Methods using combination of motion and shape features 9
CHAPTER 3: PROPOSED METHODOLOGY **11**
3.1 Skeleton Extraction 11
3.2 Pre Processing 12
3.3 Feature Extraction 12
3.4 Post Processing 13
3.5 Activity Recognition 13
3.5.1 Feed Forward Back Propagation Neural Network 14
3.5.2 K Nearest Neighbours Classifier 14
3.5.3 SVM Classifier 15
3.5.4 Linear Discriminant Classifier 16
3.5.5 Naïve Bayes Classifier 17
3.5.6 Phasesof Activity Recognition 17
CHAPTER 4: IMPLEMENTATION, RESULTS AND DISCUSSION **19**
4.1 Dataset Collection 19
4.1.1 MSR Daily Activity 3D Dataset 19
4.1.2 Le2i Fall Detection Dataset 20
4.2 Network Architecture 21
4.2.1 Back Propagation Neural Network 21

4.2.2 K Nearest Neighbours 21
4.2.3 SVM 21
4.2.4 Linear Discriminant 21
4.2.5 Naïve Bayes 21
4.3 Performance Parameters 22
4.4 Activity Recognition Without Data Normalization 22
4.4.1 Classifier Results and Discussion 23
4.5 Activity Recognition With Data Normalization 25
4.5.1 Classifier Results and Discussion 26
4.6 Comparison with State of the Art 28
CHAPTER 5: CONCLUSIONS AND FUTURE WORK 31
APPENDIX A 32
REFERENCES 38

List of Figures

Figure 3.1: Human Skeleton obtained through Openpose 11
Figure 3.2: System Framework 13
Figure 3.3: Flowchart for Activity Recognition Process 18
Figure 4.1: Sit,Stand and Walk Samples from MSR Daily Activity Dataset 19
Figure 4.2: Fall Samples from Le2i Dataset 20
Figure 4.3: Confusion matrix for Knn without normalization 23
Figure 4.4: Confusion matrix for SVM without normalization 23
Figure 4.5: Confusion matrix for LDA without normalization 24
Figure 4.6: Confusion matrix for NB without normalization 24
Figure 4.7: Confusion matrix for BPNN without normalization 25
Figure 4.8: Comparison of accuracy without data normalization 25
Figure 4.9: Confusion matrix for Knn with normalized data 26
Figure 4.10: Confusion matrix for SVM with normalized data 27
Figure 4.11: Confusion matrix for LDA with normalized data 27
Figure 4.12: Confusion matrix for NB with normalized data 27
Figure 4.13: Confusion matrix for BPNN with normalized data 28
Figure 4.14: Comparison of accuracy with data normalization 28

List of Tables

Table 4-1: Number of frames used for training and testing 20
Table 4-2: Performance comparison without data normalization 23
Table 4-3: Performance comparison with data normalization 26
Table 4-4: Comparison with the state of the art for class Fall 29
Table 4-5: Comparison with the state of the art for classes Walk and Sit 30
Table 4-6: Comparison with the state of the art for class Stand 30

CHAPTER 1: INTRODUCTION

Human activity recognition in images and sequence of images is an interesting and challenging area of research that has been attracting numerous researchers for the past two decades. It is an important step towards behaviour understanding, which can be utilized in the fields of automated surveillance and monitoring, and building smart and intelligent environments for assisted living.

1.1 Motivation, Scope and Background

The goal of human activity recognition (HAR) systems in images and sequence of images is to examine a video stream to extract useful information including spatial, temporal and environmental that can help in understanding human behaviour and interpreting the ongoing events. This task, which is traditionally performed by human operators, is now being replaced by machine vision based HAR systems that make use of latest technologies to automatically identify human actions in a video. HAR using machine vision and machine learning techniques has an advantage over manual systems because events of interest are rare occurrences in a video stream. Manual systems require more labour and constant vigilance, which makes them tiresome and inefficient. HAR systems can take away the element of human error and are faster and cost effective. However, processing video data to extract useful information and making adequate decisions is a challenging task due to the high dimensionality of video data, background noise and interconnected actions and interactions of various objects.

The process of human activity recognition in images and image sequences involves two major parts. (a) Action representation using features extracted from the video stream and (b) Action classification. There are many methods used for activity recognition using various features. These approaches can be categorized depending upon the types of features/method used for action representation.

1.1.1 Categories of HAR

Five main approaches based on action representation methods are:

a) Space-Time Approaches: These methods use spatiotemporal features for action representation. These represent activities as 3D space-time volumes. Features include optical flow, motion vectors, trajectories, speed and direction. [1][2]

b) Rule based Approaches: These approaches use semantic features to represent activities. Actions are defined using descriptive models based on a set of rules or attributes. [3]

c) Shape based Approaches: In these approaches, silhouettes or skeleton are extracted and the activities are represented using features like contours, centroid of the silhouettes, joint angles and distances etc. [4]

d) Stochastic Approaches: These methods use statistical models like Hidden Markov Models to represent activities as a sequence of predictable states. [5]

e) Multimodal: These approaches combine various types of features to get a better understanding of the ongoing scene in the video like combining depth data with the RGB data [6]. Some approaches may combine the outputs of visual and non-visual sensors for example by combining audio data with the motion features etc. [7]

For action classification two approaches are used:

(a) Machine learning based Approaches: These approaches can be further classified as supervised and unsupervised learning techniques. Supervised learning is used for labelled data. The classifier tends to assign a class to new or unknown data based on the similarity between new and train data. Widely used supervised learning techniques include artificial neural networks, SVM, KNN etc. [8]. Unsupervised learning techniques are used when the labels for a given dataset are not known. Statistical models are applied on features that are extracted from unlabelled video data. Common unsupervised learning techniques include K means clustering, Gaussian mixture models, HMMs etc. [9]

(b) Template matching based Approaches: In these approaches, activities are represented through templates and action recognition is performed by template matching. Templates can be based on various features like pose, motion history etc. [10]. In some approaches, images are decomposed in to small ROI. Each region then act as a feature. In this case, no human localization is required. [11]

1.1.2 Applications of HAR

Activity recognition systems have broad scope with a wide range of applications. These include:

1) Content-based video retrieval/video indexing: these systems help users to search for a video in large databases (e.g. internet videos). User can present a query to the retrieval

system. System matches the query with the database videos and returns a result if found any. [12]

2) *Robotics:* It's important to learn human actions for an autonomous mobile robot for example a cleaning robot which has to navigate without any human supervision in an uncontrolled environment [13]
3) *Human computer interaction:* Action and gesture recognition for enhancing ways for human computer interaction
4) *Ambient Assisted Living:* these monitoring systems are useful for tele rehabilitation and telecare for elderly. [14]
5) *Ambient Intelligence:* Ambient Intelligence systems are a kind of AI systems that make our environment smart and sensitive to human presence and actions. These systems can sense human presence, their actions and surroundings and based on these can perform a series of actions that benefits the human to make life much easier e.g. in intelligent homes. [15]
6) *Visual surveillance:* Action recognition in visual surveillance is helpful in categorizing normal and unusual activities going on in the environment so that to reduce crime rate. [16]

1.2 Research Objectives

Most of the recent research on activity recognition is based on the data obtained from RGB-D cameras like Kinect device that gives depth information along with the human skeleton information. These researches, even though report high accuracy results but are device dependent and hence cannot be applied to general 2D CCTV camera videos. Current research focuses on using only 2D skeleton information for activity recognition. Another objective is to identify the best performing classifier by comparing the performance of some widely known classifiers. Applications related to video retrieval and surveillance can greatly benefit from this research.

1.3 Thesis Organization

Rest of the thesis is organized as follows. Chapter 2 gives a detailed description about the various types of methodologies used for feature extraction and action recognition in the literature. In chapter 3, proposed methodology is presented along with a brief summary of supervised learning techniques used for action classification. Chapter 4 discusses about the datasets used for the research, results obtained through classifiers and the comparison of the classifiers' performance. At the end, Chapter 5 concludes the thesis with some suggestions for future work.

CHAPTER 2: LITERATURE REVIEW

The field of vision based activity recognition has seen many advancements in recent times. Many researchers have used various techniques for activity recognition. In different researches, data from different modalities including single camera, stereo and infrared has been used. Some of the work from recent years related to the current research has been discussed here.

2.1 Methods using skeletal features

Manzi et al. (2017) use the concept of "key poses" extracted from the skeletal data to recognize a human activity. They present the idea that a small number of key poses or informative postures is sufficient to describe an activity. Moreover they use dynamic clustering during the classification phase to find optimal number of clusters (key poses) for the given input sequence instead of using a fixed or predefined number of clusters. They extract 3D positions of skeletal joints using a depth camera and perform a normalization step to make the raw data independent of sensors position and person size. Number of informative postures required to describe an activity are found using clustering so that similar postures are grouped into clusters sharing similar features. The centroids of these clusters then represent key poses. They apply K means clustering algorithm multiple times on the input sequence using different values of k so that multiple samples representing the same activity are generated. Ordered pair of centroids represent the activity. In final step of feature extraction, the authors discard equal and consecutive centroids in the temporal order and only consider transition between centroids. From this single sequence, many overlapping small n-tuples are generated by considering a short sliding window, where each instance is assigned a weight which increases for each repeating tuple in the sequence. Hence each activity is represented by many new activity features generated from different sets of clusters. Using these activity features, training is done using a multiclass SVM classifier which associates each activity feature set with an activity and several models. For testing, dynamic clustering is used to find optimal number of clusters using X means algorithm. Activity features extracted from the test sequence along with previously trained model relative to these clusters are used together to classify the activity. They have reported quite good performance on 100 frames of data from CAD-60 dataset with around 98% accuracy, which reaches to almost 100% when using 500 frames [17].

Le et al. (2013) have used a Kinect to obtain 3D positions of 20 joints in space. From these coordinates they calculate joint angles for certain selected joints. They perform seven various experiments using different number of joints, with, and without data scaling to compare the system's performance. They detect four postures, sitting, standing, bending and lying using SVM classifier. They have prepared their own dataset for training and testing purpose. Their results show that best results (98.6%) were obtained using 9 joint angles for posture detection [18].

Cippitelli et al. (2016) combine skeletal and depth features for activity recognition. Kinect is used to obtain 3d positions of the joints. For normalizing the raw data obtained from the Kinect sensor, they divide the distance of a joint *i* with the torso joint to the distance between neck and torso joints. This step is performed to make the system invariant to the camera position with respect to the test subject. Feature vector containing normalized distances of all the joints are calculated in this way. Then they use K means clustering technique to group all frames of an activity into similar clusters representing the key poses in an activity. For final classification step they use SVM classifier. Their system works with an accuracy of 95% on KARD dataset and 93% on CAD-60 dataset [19].

Li et al. (2016) have used two depth based features, Local Occupancy Patterns (LOP) and Histogram of Oriented Principal Component (HOPC) along with skeletal features to recognize activities. These depth features are based on the 3D point cloud around a joint. They apply Multiple Features Sparse Fusion (MFSF) on the three types of features to obtain final feature vector. They use SVM for activity classification. They report 95.6% accuracy on MSR Daily activity dataset and 94.3% accuracy on MSR Action 3d dataset [20].

2.2 Methods using silhouette based features

Kushwaha and Srivastava (2016) have applied approximate median filter based model for foreground segmentation. They extract distance signal feature based on contour points of the human silhouette for various poses like sitting, walking etc. Firstly they extract the contour of the silhouette and then they calculate centre of mass of the silhouette. Distance signal is generated by finding distance between each contour point and centre of mass. For activity classification, they have used an SVM classifier. They have tested their technique on KTH and WVU multi-view datasets. Their system works with an 88.5% accuracy [21].

Zerrouki et al (2018) use pose feature extracted from human silhouettes for activity classification. They first perform background subtraction for segmentation of human body in the image. Then they extract pose feature from the pixels constituting the silhouette area. They partition the body area into five occupancy zones corresponding to the areas occupied by head, arms and legs. Then they calculate five ratios of individual areas with the total silhouette area. For action classification they use AdaBoost classifier based on several weak classifiers to recognize six action classes i.e. sitting, standing, bending, lying, kneeling and squatting. They have used Decision Stump as the weak learner. They report an overall accuracy of 96.56% on URFD dataset and 93.91% accuracy on UMAFD dataset [22].

Goudelis et al. (2015) have used Trace transform to extract spatiotemporal features for fall detection in a video stream. They first extract silhouettes from the frames and then apply trace transform on each frame. Then a diametric functional P is applied on the columns of the trace transform. Then another functional Φ is applied to get the triple features. The final feature vector is trained using SVM classifier [23].

2.3 Methods using motion features

Mu et al. (2016) have extracted motion vectors directly from the video streaming for fast computation. They use an adaptive threshold method for moving object segmentation. They normalize all macroblocks as 4x4 and convert all reference frames of motion vectors to previous frames. Then modulus of motion vectors is obtained. Adaptive threshold value is obtained by iterating all the individual modulus values which would make between class variance maximum. Considering that {MV1, MV2,...MVs} are the values of motion vectors then for MVi ($0<i<s+1$) , they find probabilities of two types of motion vectors. One with values less than MVi and others with values greater than MVi. Mean and between-class variance for both types is calculated and optimal threshold value is found. The moduli of motion vectors are segmented using this value and hence moving target region is obtained. They calculate velocity and direction of motion vectors and use it to obtain six features including average direction and velocity of the target, variance of direction and velocity and entropy of direction and velocity. Apart from these intra-frame features, an inter-frame feature is also extracted based on the Histogram of motion vectors in moving target region for each frame. If histogram of two consecutive frames is small then this feature, referred to as $inter_D$, is also small and vice versa. Lastly they use SVM classifier to classify

between normal and abnormal behaviour. Four classes of abnormal behaviour are classified including wandering, following, chasing and falling down. They have built their own database for five classes. Their system works with 91.7% accuracy on their own dataset [24].

The approach adopted by Xia et al. (2015) is to solve a new problem based on a previous similar problem. They present a saliency based visual attention model to segment a complex behaviour into sub-behaviours constituting a single action. Then target behaviour is detected based on similarity between sub-behaviours in detection videos and behaviour cases.

They use motion vectors for behaviour decomposition. They calculate motion vectors using ARPS algorithm. Then they extract certain features from these motion vectors including motion intensity and motion orientation consistency based on histogram of orientations and entropy. By combining these features they obtain motion saliency maps. They define saliency value in each frame as the average brightness of attended regions in motion saliency map. If the difference of saliency value of any two consecutive frames is larger than a threshold, behaviours are different so that frame is used as behaviour segmentation point in the video.

Sub-behaviours are composed of the actions of individual body parts. The authors use hidden Markov models to detect actions of individual body parts including head, upper body, arms and lower body and then use forward/backward algorithm of HMMs to record start and end time of each actions. Then a sub-behaviour representation using context free grammar is constructed based on individual body parts' actions and their relationships. Behaviour is then represented using sequences of sub-behaviours and their corresponding time duration in order. Hence a complex behaviour is decomposed into a sub-behaviour sequence and a duration time sequence. From sub behaviour sequence, they calculate frequency of occurrence of a sub-behaviour and call it sub behaviour attribute characteristics and from duration time sequence, they calculate total time of occurrence of a particular sub behaviour and call it time attribute characteristics. Finally they match the detected behaviour with the behaviours in case database. If a behaviour matches with a similar behaviour as in case database, it is recognized as suspicious behaviour. To find similarity in sub behaviours, they calculate cosine of eigenvectors of sub behaviours attribute characteristics of behaviours in case base and detected videos. Then they measure similarity in temporal order and time duration of sub behaviours using an order factor and a span factor. They calculate order factor using maximum number of longest common sub sequence and total number of sub behaviours in a specific case and a video. To calculate span factor, they calculate time duration

difference in both videos. For testing, they used CAVIAR and BEHAVE datasets. They detected four types of activities i.e. fighting, chasing, loitering and fainting [25].

2.4 Methods using template matching techniques

Wachs et al. (2010) have devised a human posture recognition system in context of pedestrian behaviour prediction for intelligent vehicle systems. They used template matching technique to match features in the labelled and unlabelled data. They detect 8 body poses including 4 views of standing and 4 views of kneeling postures. In their approach, they create a dictionary of human body patches selected from inside of an annotated silhouette. They take eight images for each class and convolve them with a delta function, a Gaussian and x and y derivatives. Then they select 20 patches from each resulting image. Patch and its location with respect to the centre of the object is stored in a dictionary entry along with the applied filter. Hence making 640 entries per class. To obtain feature vectors for training images, they convolve the images with the filter in the dictionary entry and then cross correlation with the patch in the same dictionary entry yields a strong response where the patch appears in the filtered image. Then they apply 1D filters representing the location of the patch in the dictionary entry to the cross-correlated image to obtain voting for the object centre. By this method, they obtain a training set of 200 positive and 4000 negative samples each with 640 features. Then they use multiclass AdaBoost algorithm with shared features to add weak learners to produce a strong classifier for training. Each weak classifier contributes to improve the overall classification rate. For object detection a score for strong classifier is calculated based on votes from weak learners. High values in the voting array indicate that weak learners agreed on the centre of the object. At the end, non-maxima suppression is applied to find peaks in the voting array. Highest maxima determines the class. They use high level classifiers to enhance the output of the low level classifier for error correction. High level classifiers take as input the class labels which are output from the low level classifier. They use HMMs as higher level classifiers for error correction. Their reported accuracy for marines' detection is 98.7% for uncluttered marines and 53% when there is occlusion due to clutter [26].

2.5 Methods using combination of motion and shape features

Wang et al. (2014) use combined motion and appearance features from untrimmed videos for action recognition. They firstly segment the videos using a temporal sliding window of 150

frames to obtain short video clips. Then they perform action recognition independently on these video clips. Result of the action recognition of full video is based on the combined results of these short clips. The extracted motion features include four descriptors including HOG, HOF, MBHx and MBHy. For dimensionality reduction, PCA is used. These local descriptors are then combined using Fisher Vector Representation. Appearance features are extracted using Convolutional Neural Networks. 15 frames from each video clip are selected and a 4096 D feature vector is extracted from each frame. An average pooling is done to obtain a global representation for the full video clip. Both motion and appearance features are then combined to generate a single final feature vector for the video clip. Multiple one-vs-all SVM classifiers are trained for each action class using the combined motion and appearance feature vector. Based on the classifier predictions for each video clip, prediction for the whole sequence is generated by defining two threshold values. $\tau 1$ for clips and $\tau 2$ for video, where threshold represented the maximum number of action classes. These threshold were used to eliminate results of the predictions with low SVM scores. Their results showed that action recognition was sensitive to the selected threshold values [27].

Albawendi et al. (2018) combine motion and shape features for fall detection in home environment. First they perform background segmentation to obtain human silhouette. Then they obtain motion features based on motion history image. Then they fit an ellipse around the human body to find the change in shape during the fall event. They have assumed that for a fall in a video, motion is larger with higher acceleration as compared to the other normal activities. The combined features include the change in the orientation of the ellipse and rate of change of human motion. Another feature known as projection histogram is also calculated. They calculate the vertical and horizontal projection histograms for each activity and then obtain the difference between the maximum values of both types of histograms. They report 99% accuracy on their own dataset collected in a home environment using seven people [28].

CHAPTER 3: PROPOSED METHODODLOGY

This chapter gives a detailed description of the proposed methodology for activity recognition in videos. Activity recognition is performed based on 2D joints' position extracted directly from the video sequence. Two types of features are extracted using skeletal data. (a) Shape features – including angles and distances between the skeletal joints. (b) Motion features – joint motion in consecutive frames.

3.1 Skeleton Extraction

Human skeleton provides great deal of information about the human posture in a video frame. This posture information can be combined with the motion information for distinguishing various human actions.

In this study, positions of body joints were extracted using openpose library [29]. Openpose is an open source library that is based on the works of Cao et al (2017) that takes an image or video sequence as input and produces an output containing the locations of 18 human skeletal joints as key-points [30].

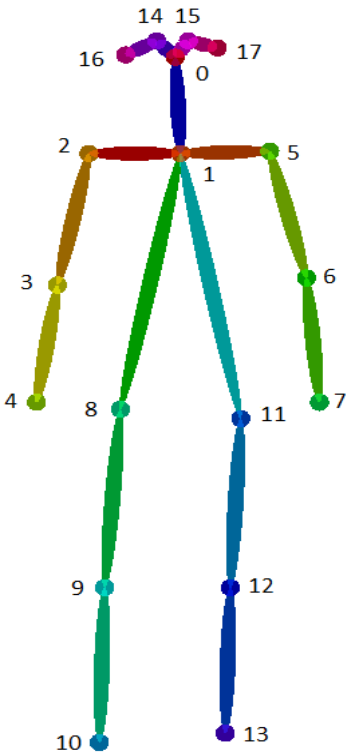


Figure 3.1: Human Skeleton obtained through Openpose

3.2 Pre Processing

The Openpose library gives locations of the body joints of every person in the frame. The output is stored in a yml file containing person number, joint number and 2d positions of joints respectively. A separate file with the joints information is stored for each frame. It was observed that the order in which the persons' key points are stored can change in some frames of any video. Hence, in order to differentiate the key points of the person in the foreground i.e. the actor, from the person/s in the background, a pre-processing step was performed which extracts the key-points of only the actor and discards the positions related to people in the background.

3.3 Features Extraction

Openpose gives total 18 Skeletal Joints positions as key-points out of which eight active joints were selected for this research including left Shoulder, left hip, left knee, left ankle, right shoulder, right hip, right knee and right ankle, considering these to be more relevant to the types of activity we want to recognize. The positions of the selected joints were used to extract shape and motion features for action recognition. These features included:

1. angle between left hip and left knee
2. angle between right hip and right knee
3. angle between left knee and left ankle
4. angle between right knee and right ankle
5. angle between left hip and left ankle
6. angle between right hip and right ankle
7. ratio of distances between left hip, left knee and left knee, left ankle
8. ratio of distances between right hip, left knee and right knee, right ankle
9. displacement between consecutive frames of right shoulder
10. displacement between consecutive frames of right hip
11. displacement between consecutive frames of right knee
12. displacement between consecutive frames of right ankle
13. displacement between consecutive frames of left shoulder
14. displacement between consecutive frames of left hip

15. displacement between consecutive frames of left knee

16. displacement between consecutive frames of left ankle

These 16 types of features were extracted for each frame of the video sequence. Thus the resulting feature vector was an array of (total number of frames in the video) x 16 entries.

3.4 Post Processing

A window of 10 frames was used for averaging the output of feature vector to remove any noise in case of faulty detection. The final feature vector then became an array of (total number of frames in the video/10) x 16 entries. This feature vector was then input to the classifier, for classifying the activity into one of the four classes i.e. sit, stand, fall or walk. A complete framework for the proposed system is presented in figure 3.2.

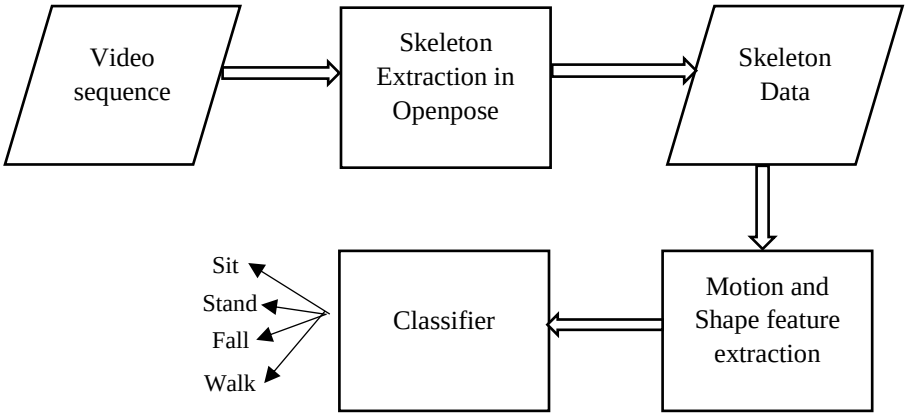


Figure 3.2: System Framework

3.5 Activity Recognition

The recognition step was performed using supervised machine learning. Performance of the selected classifier remains a major factor in the final performance of the algorithm, hence it is of prime importance to select the best performing classifier for a particular method. Five widely used classification techniques were used for classification of activities in to four classes. The results of all the techniques were compared with each other and with the state of the art to identify the best performing technique for the proposed method. A brief summary of each technique is given here.

3.5.1 Feed Forward Back Propagation Neural Networks

Feed Forward neural networks are a powerful tool for pattern recognition problems. These are also known as Multilayer Perceptron. To learn a function $y=f(x)$, these networks define a mapping $y=f(x,\theta)$ and approximate the function by learning the values of parameters θ. Feed forward refers to the forward flow of information from input to output i.e. there are no feedback connections. There are three types of layers in these networks. Input layer, in which the input data is received. Output layer, which gives the output of the network. Hidden layer/s, the layers that lie between input and output layers. These are called hidden because the output of these layers is not directly specified by the training data. Training data contains example inputs x with an accompanying label y approximating the output for the particular input. The hidden layer is composed of many units that act in parallel and function similar to a neuron. Working of the neurons in hidden layers mimics brain function. These units take input from other units and compute their own values for activation called weights. Various functions are used to calculate these activation values and are known as activation functions. Activation functions decide whether a neuron will fire for a particular input or not. Commonly used activation functions are sigmoid function, hyperbolic tangent, softmax and rectified linear unit (ReLU) [31] [32].

Back propagation network is an important type of feed forward neural networks. In these networks, gradient descent method is used to minimize the total error of the output i.e. error is propagated backwards. For detailed description about back propagation networks [33] [34] can be referred.

3.5.2 K Nearest Neighbours Classifier

K nearest neighbours or simply knn is a supervised learning technique, which uses example data that is separated into various classes to predict the class of a new sample. A model is constructed based on the relationship between predictors and targets from the training data. The model is then used to determine the class of data with unknown labels. To learn the class of a new instance, similar instances from the training data are found, known as neighbours. The class of the new sample, which is most common amongst its *K* nearest neighbours, is assigned through majority voting. Similarity between data instances is found by using a distance function. The example of some commonly known distance functions are Euclidean, Minkowski, Mahalanobis, Manhattan etcetera. The detailed description of the algorithm can be found in [35] [36]. The

Minkowski distance between two points a and b in a d dimensional space is given by [36]:

$$L_p(a,b) = (\sum_{i=1}^{d} |a_i - b_i|^p)^{\frac{1}{p}} \quad (3.1)$$

When p= 1, the distance is called city block distance or Manhattan distance. For p=2, the Minkowski distance is equivalent to Euclidean distance. The Mahalanobis distance between two points x and y is given by [37]:

$$d_M(x,y) = \sqrt{(x-y)^T S^{-1}(x-y)} \quad (3.2)$$

Where S is the covariance matrix.

3.5.3 SVM Classifier

SVM classifier finds an optimal hyperplane that maximizes the separation between classes. Support vectors are the data points that lie closest to the hyperplane and have maximum impact on its location. Out of many possible hyperplanes, SVM finds the most optimal one. Hence finding the optimal hyperplane is an optimization problem that is solved by general optimization techniques. The distance between the hyperplane and the closest data point (support vector) is called the margin of separation. Optimal hyperplane is the one that maximizes this margin. For problems that are not linearly separable, SVM uses kernel functions that transform low dimensional input space to higher dimension so that to convert these in to separable problems.

Assuming a linearly separable case of binary classification, say we have some training data $\{x_i, yi\}$ where x_i is the input and y_i are the associated labels. A hyperplane separating the data into positive and negative samples will have equation of the form:

$$H_0\text{:-}\ \mathbf{x_i.w} + b = 0 \quad (3.3)$$

Here w is the normal to the hyperplane known as the weight vector and b is the bias. Defining two hyperplanes as:

$$H_1\text{:-}\ \mathbf{x_i.w} + b \geq +1 \text{ for } y_i = +1 \quad (3.4)$$

$$H_2\text{:-}\ \mathbf{x_i.w} + b \leq -1 \text{ for } y_i = -1 \quad (3.5)$$

Where the points on H_1 and H_2 are the support vectors. The two equations (3.4) and (3.5) can be combined as:

$$y_i\ (\mathbf{x_i.w} + b) \geq 1 \quad (3.6)$$

If d+ is the shortest distance to the nearest positive sample from the separating hyperplane H_0 and d- is the shortest distance of the nearest negative sample, then the distance d+ + d- is called the

separating margin and the task of SVM is to maximize this margin. The geometric margin between the planes H_i and H_2 is 2/||w||. Hence the optimization problem becomes: minimize $||w||^2$ subject to the constraint given by equation 3.6. This quadratic problem can be solved using Lagrangian Multipliers. The Lagrangian for SVM is [38]:

$$L_p = \frac{1}{2}||\boldsymbol{w}||^2 - \sum_{i=1}^{l} \propto_i y_i(\boldsymbol{x_i}.\boldsymbol{w} + b) + \sum_{i=1}^{l} \propto_i \quad (3.7)$$

By applying the derivative to find minimum, the following conditions are obtained [38]:

$$w = \sum_i \propto_i y_i x_i \quad (3.8)$$

$$\sum_i \propto_i y_i = 0 \quad (3.9)$$

These can be substituted in equation 3.7 to get the dual formulation[38]:

$$L_D = \sum_i \propto_i - \frac{1}{2}\sum_{i,j} \propto_i \propto_j y_i y_j \boldsymbol{x_i}.\boldsymbol{x_j} \quad (3.10)$$

The problem is now to maximize equation 3.10 with respect to alpha subject to the constraint given in equation 3.9. For further details on SVM [39] can be referred.

3.5.4 Linear Discriminant Classifier

LDA is one of the many possible techniques for the classification problem. LDA seeks for linear transformations that best separate the data while preserving the useful information. It takes into account inter class and intra class scatter. Fisher's linear discriminant finds projection of the data onto a lower dimensional subspace that maximizes the ratio of between class to within class variance.

Assuming we have x_i samples belonging to two classes, then the projection of these samples on a line given by unit vector w can be given as $w^t x_i$. The criterion function for Fishers discriminant is given as [36]:

$$J(w) = \frac{w^t S_B w}{w^t S_W w} \quad (3.11)$$

Where S_B is the between class scatter matrix and S_W is the within class scatter matrix for the given data. The w, which maximizes the criterion function J(w), gives the best separation between the classes. By taking the derivative of J, we get the general eigenvalue equation as [36]:

$$S_W^{-1} S_B w = \lambda w \quad (3.12)$$

Between class scatter is defined in terms of the class means. Since only the direction of w is important and not the magnitude and $S_B w$ is always in the direction of (m_1-m_2) where m_1 and m_2 are the means of classes, the solution for w can be directly written as [36]:

$$w = S_W^{-1} (m_1 - m_2) \tag{3.13}$$

For multiple classes, the generalized form of LDA is given in terms of the transformation matrix **W** as [36]:

$$J(W) = \frac{|\boldsymbol{W}^t S_B \boldsymbol{W}|}{|\boldsymbol{W}^t S_W \boldsymbol{W}|} \tag{3.14}$$

3.5.5 Naïve Bayes Classifier

The Naïve Bayes classifier predicts the classes of unknown data using Probability theory and Bayes theorem. It calculates the probability of a data point belonging to each class and assigns the class with the highest probability to the data point. The algorithm assumes that every feature is independent of the other features in a class and independently contribute to the probability of the class hence the name naïve.

Bayes Theorem: Say we have some data x belonging to two classes with labels C (C=0 or C=1). The equation for Bayes Theorem is given as:

$$P(C|x) = \frac{P(C)p(x|C)}{p(x)} \tag{3.15}$$

Here P(C|x) is known as the posterior probability. P(C) is called the prior probability of C belonging to a certain class regardless of the value of x. p(x|C) is a conditional probability known as class likelihood. It relates to the probability of the attribute having the observation values x for a given class C. p(x) is the evidence that is the probability of an observation x regardless of its class [40].

3.5.6 Phases of Activity Recognition Process

Activity recognition process was composed of two phases.

1. *Training Phase*: In training phase, the feature vectors for all the training videos were extracted separately and then all the feature vectors were combined to form the final train vector which is stored as a separate file. Labels for the train vector were also stored in a separate file. These files could later on be accessed by the program when any test video was presented.

2. *Testing Phase:* In the testing phase, the feature vector for the query video was extracted. The train feature vector, corresponding labels and the test feature vector were input to one of the classifiers (section 3.5.1 - 3.5.5). The classifier then generated a model for the test vector and labeled the video accordingly. Flowchart for the activity recognition process is presented in figure 3.3.

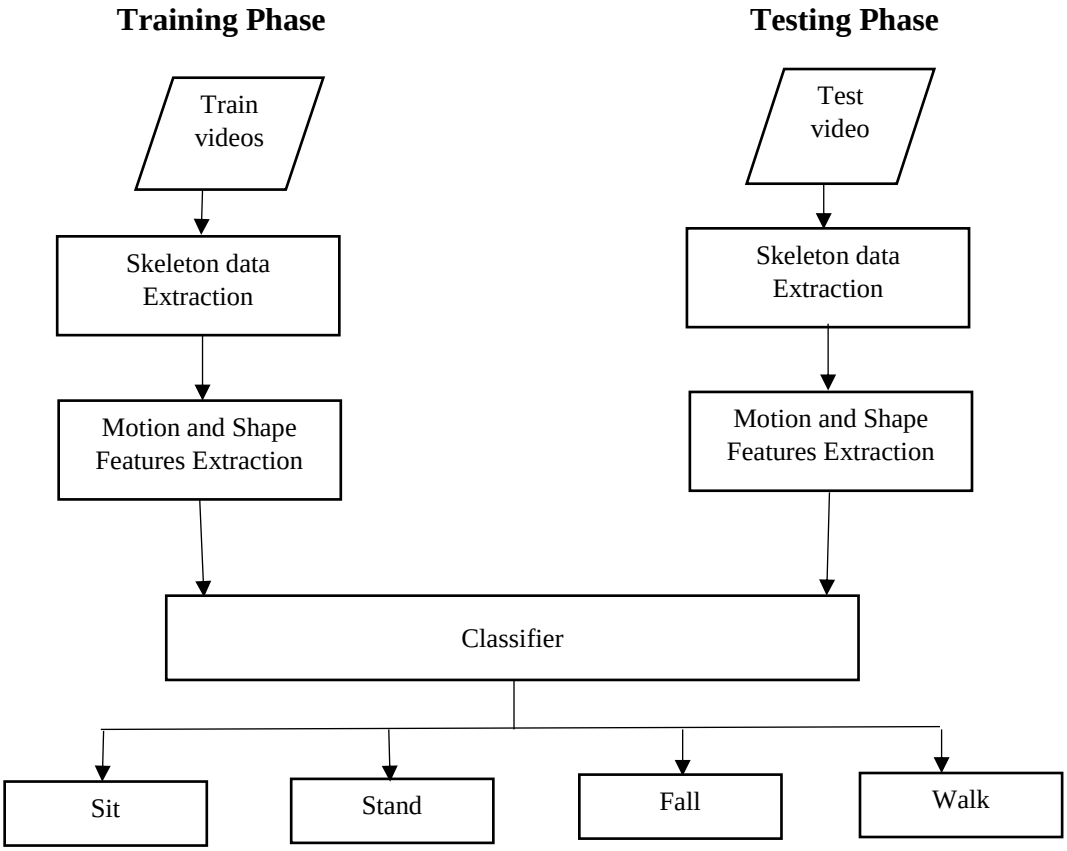


Figure 3.3: Flowchart for Activity Recognition Process

CHAPTER 4: IMPLEMENTATION, RESULTS AND DISCUSSION

This chapter discusses about the implementation details for the proposed method and the evaluation criteria to measure the performance of the algorithm. At the end, the classification results of the proposed method are compared with the state of the art.

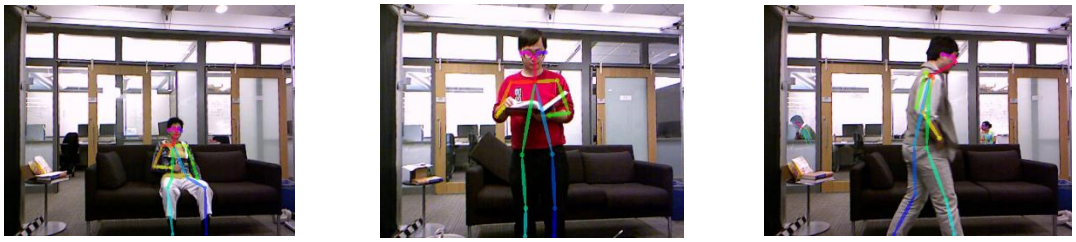
4.1 Dataset Collection

To evaluate the performance of the algorithm, two publically available dataset were used.

4.1.1 MSR Daily Activity 3D Dataset [41].

This dataset contains videos of 10 people performing various activities in sitting and standing positions like eating, drinking, using laptop, talk on phone etc. The videos are captured using a single fixed angled camera in a living room. The resolution of the dataset is 640x480 pixels. Total 68 videos from this dataset for three classes with labels sit, stand and walk were used in this research. Some samples from this dataset before and after the extraction of skeleton are shown in figure 4.1.

(a)



(b)

Figure 4.1: Sit, Stand and Walk Samples from MSR Daily Activity Dataset
(a) before skeleton extraction (b) after skeleton extraction

4.1.2 Le2i Fall Detection Dataset [42]:

The videos are recorded in four different locations using a single camera including home, office, lecture room and coffee room. The video sequences are recorded using variable illumination and cluttered and textured background. The actors fall while performing various day-to-day activities. The resolution of the dataset is 320x240 pixels. Total 30 videos from this dataset were used in this research. Some samples from this dataset before and after the extraction of skeleton are shown in figure 4.2.

(a)

(b)

Figure 4.2: Fall Samples from Le2i Dataset

(a) before skeleton extraction (b) after skeleton extraction

Total 16520 frames from 98 videos were used in this research. The distribution of data sets' frames for training and testing is given in table 4-1.

Table 4-1: Number of frames used for training and testing

Dataset	Total Frames	Train Frames	Test Frames
MSR	12540	6070	6470
Le2i	3980	2250	1730
Total	16520	8320	8200

4.2 Network Architecture

Implementation details of the machine learning techniques used for action recognition are given in this section.

4.2.1 Back Propagation neural network:

The architecture used in this research was implemented using Matlab nprtool. It was composed of a single hidden layer with 14 hidden neurons. The distribution of input data was such that 50% samples were for training, 15% were for validation and 35% were for testing purpose. The Activation function used was sigmoid for hidden Layer and softmax for output layer.

4.2.2 K Nearest Neighbours:

The architecture of KNN was implemented using Matlab fitcknn with k=4. The value of k was found by adjusting the value until the classifier had lowest error. Mahalanobis distance had been chosen as the distance metrics to find k closest points.

4.2.3 SVM:

The SVM Classifier was implemented using Matlab ECOC model which uses SVM learners. This model reduces multiclass problems to a series of binary classifiers using a one versus one coding design. In a one versus one coding design, for each binary learner, one class is considered positive, another is considered negative and the rest are overlooked. It uses all possible class pair combinations for learning.

4.2.4 Linear Discriminant:

The architecture for linear discriminant was implemented using Matlab fitcdiscr. This function takes training data and their labels as input and outputs a discriminant analysis model which is then applied on the test samples to predict labels.

4.2.5 Naïve Bayes:

Naïve Bayes classifier was implemented using Matlab fitcnb. The model parameters i.e. mean and standard deviations of predictors in each class were estimated based on Gaussian distribution.

4.3 Performance Parameters:

The performance evaluation parameters for the classifiers included accuracy, precision and recall. The formulae are given in equations 4.1-4.3.

$$\text{Accuracy} = \frac{TP+TN}{P+N} \tag{4.1}$$

$$\text{Precision} = \frac{TP}{TP+FP} \tag{4.2}$$

$$\text{Recall} = \frac{TP}{TP+FN} \tag{4.3}$$

Here TP = True Positive, TN = True Negative, FP= False Positive, FN=False Negative, P = Total positives, N = Total negatives.

The results of the activity recognition on the selected datasets were obtained by considering two scenarios. First, using the original feature vector values and second, after applying data normalization to obtain the values from [-1, 1] range, using the formula in equation 4.4, for angle values and [0, 1] range, using the formula in equation 4.5, for displacements and ratios. Since the resolution for both the dataset used is different, this step helps in making the data more comparable by converting the values to a similar range.

$$(x - x_{\text{min}})/(\text{x}_{\text{max}} - \text{x}_{\text{min}}) \tag{4.4}$$

$$2 \text{ x } (x - x_{\text{min}})/(\text{x}_{\text{max}} - \text{x}_{\text{min}}) - 1 \tag{4.5}$$

4.4 Activity Recognition without Data Normalization

First set of experiments included collection of the results for original values of feature vector i.e. without applying any normalization to the test and train data. Table 4.2 gives the results for all the classifiers. Highest overall accuracy for action classification using original features' values was observed using Naïve Bayes classifier with 90% test videos correctly classified. Comparison of the recall values show that the class Fall was best identified using LDA. For class Walk, LDA and NB both show better recall than others. NB and KNN both showed a 100% recall value for the class Stand. For Sit best value is achieved using BPNN. It was also observed that even though recall for sit is relatively less in all the classifiers, precision is very high which means very few samples of the other classes were misclassified as sit however almost all the false negatives from the class sit were misidentified as fall hence precision for the class Fall was lowest

in all the classifiers. Lowest overall accuracy was observed in the SVM classifier with a value of 76%. Detailed results in the form of confusion matrices are given in figures 4.3 - 4.7.

Table 4-2: Performance comparison without data normalization

Classifier	Fall		Walk		Stand		Sit		Overall Accuracy (%)
	Precision (%)	Recall (%)	Precision (%)	Recall (%)	Precision (%)	Recall (%)	Precision (%)	Recall (%)	
KNN	92	86	90	75	75	100	100	92	88
SVM	56	93	82	75	100	75	100	58	76
LDA	70	100	92	92	100	92	100	58	86
NB	80	86	92	92	92	100	100	83	90
BPNN	80	92	82	87	96	81	99	98	89

4.4.1 Classifier Results and Discussion:

- ***KNN:*** It can be seen from fig 4.3 that Knn gave a 100% recall value for Stand however precision for the same was the lowest. It means that most of the false negatives from the other classes were misidentified as Stand. Lowest recall value is for Walk with many test videos from Walk misclassified as Stand or Fall.

	Fall	Walk	Stand	Sit
Fall	86	7	7	0
Walk	8	75	17	0
Stand	0	0	100	0
Sit	0	0	8	92

Figure 4.3: Confusion matrix for Knn without normalization

- ***SVM:*** It can be seen from figure 4.4 that SVM gave highest recall value for Fall but precision is lowest for the same. Least accurate class remained Sit with only 58% recall.

	Fall	Walk	Stand	Sit
Fall	93	7	0	0
Walk	25	75	0	0
Stand	17	8	75	0
Sit	42	0	0	58

Figure 4.4: Confusion matrix for SVM without normalization

- ***LDA:*** LDA gave good recall results for fall, walk and stand but for sit the value remained as low as 58% with most of the sit videos misidentified as fall similar to SVM results. For LDA, again precision values for Stand and Sit were 100% which means no other sample was misclassified as these two classes.

	Fall	Walk	Stand	Sit
Fall	100	0	0	0
Walk	8	92	0	0
Stand	0	8	92	0
Sit	42	0	0	58

Figure 4.5: Confusion matrix for LDA without normalization

- ***Naïve Bayes:*** NB classifier gave highest overall accuracy with most of the test samples classified correctly. Along with accuracy, it was observed that precision and recall for all the classes were also good.

	Fall	Walk	Stand	Sit
Fall	86	8	8	0
Walk	8	92	0	0
Stand	0	0	100	0
Sit	17	0	0	83

Figure 4.6: Confusion matrix for NB without normalization

- ***BPNN:*** Backpropagation neural network gave second highest overall accuracy of 89 % as compared to other classifiers. It was observed that lowest precision was for fall which means most of the false negatives from the other classes were recognized as fall. Best precision value was for the class sit which means very few frames from any other class were detected as sit. We can also see that recall value for sit is quite high which means almost all of the sit frames were recognized correctly as compared to the other classes which have a comparatively lower recall values with stand class having the lowest recall.

Figure 4.8 presents a comparison of accuracies for all the classifiers.

	Fall	Walk	Stand	Sit
Fall	92	5	1	2
Walk	10	87	3	0
Stand	7	12	81	0
Sit	1	1	0	98

Figure 4.7: Confusion matrix for BPNN without normalization

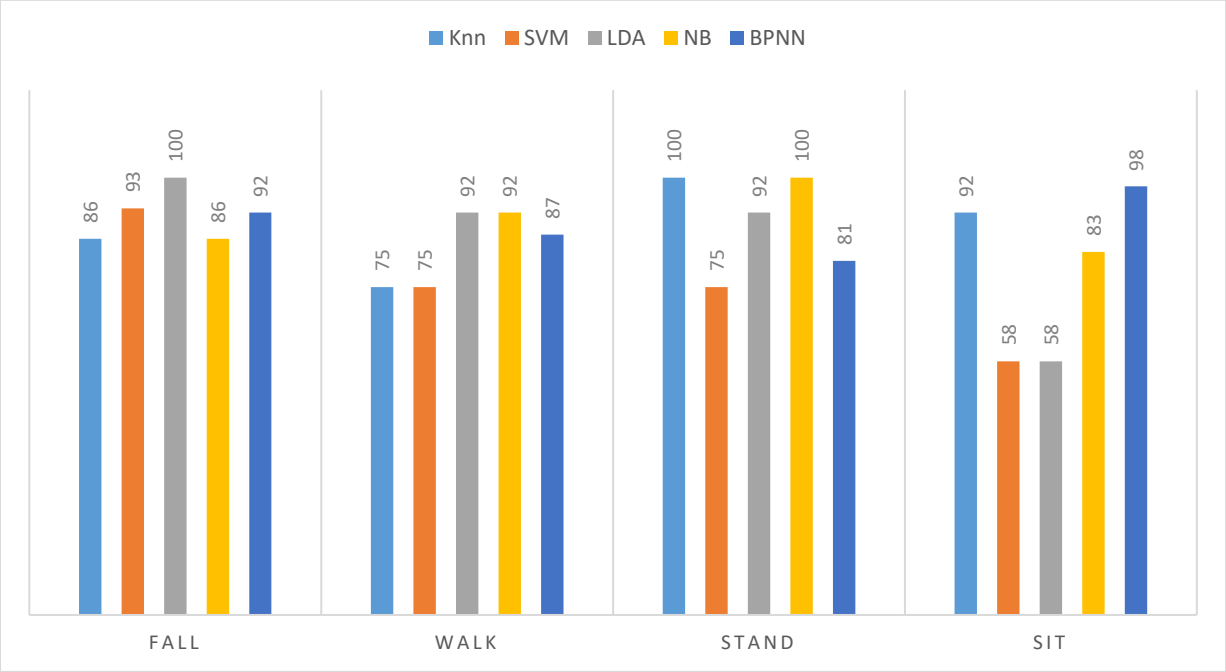


Figure 4.8: Comparison of accuracy without data normalization

4.5 Activity Recognition with Data Normalization

Second set of experiments included collection of results for all classifiers after applying normalization to the test and train data. Table 4.3 gives the results for all the classifiers. An increase in accuracy for all the classifiers was observed after applying the data normalization. Best overall accuracy was achieved in KNN where the 98% of the test videos were correctly classified. Also a remarkable increase in recall and precision was also seen for all the classifiers. It was observed that Stand, Fall and Walk had a 100% recall for KNN, SVM, LDA and NB classifiers while BPNN gave a 100% recall for Sit class. KNN also gave a relatively better recall for Sit with a value of 92% as compared to SVM and LDA which only achieved 58% recall for sit. BPNN gave good results for Sit in both sets of experiments i.e. with and without data normalization. It was also

observed that even though KNN and BPNN gave good precision for Fall, rest of the classifiers had quite lesser Fall precision value which means more test samples were misclassified as Fall as compared to any other class. Apart from KNN, BPNN and NB also gave a good overall accuracy of 96%. Detailed results in the form of confusion matrices are given in figures 4.9 - 4.13.

Table 4-3: Performance comparison with data normalization

Classifier	Fall		Walk		Stand		Sit		Overall Accuracy
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	
KNN	100	100	100	100	92	100	100	92	98
SVM	74	100	100	100	100	100	100	58	90
LDA	78	100	100	100	92	100	100	58	90
NB	87	100	100	100	100	100	100	83	96
BPNN	99.5	99	89	96	97	90	100	100	96

4.5.1 Classifier Results and Discussion:

- ***KNN:*** By the application of normalization, KNN results improved from 88% to 98%. Recall for Fall and Walk improved to a 100% from 86% and 75% respectively. Precision also increased for all the classes. By far, the best accuracy was achieved through KNN classifier. Hence, for the proposed methodology, best suited classifier was found to be KNN after the application of normalization.

	Fall	Walk	Stand	Sit
Fall	100	0	0	0
Walk	0	100	0	0
Stand	0	0	100	0
Sit	0	0	8	92

Figure 4.9: Confusion matrix for KNN with normalized data

- ***SVM:*** Although overall accuracy for SVM was greatly improved after normalization from 76% to 90%, it was still lesser than the other classifiers. Recall for all the classes improved but there was no change in the recall of Sit which remained 58% as it was before normalization. Almost half of the Sit samples were classified as Fall decreasing the precision value for Fall which was lesser as compared to the other classes whose precision was found to be 100%.

	Fall	Walk	Stand	Sit
Fall	100	0	0	0
Walk	0	100	0	0
Stand	0	0	100	0
Sit	42	0	0	58

Figure 4.10: Confusion matrix for SVM with normalized data

- ***LDA:*** Recall for Walk and Stand increased to 100% however there was no improvement in the recall for Sit which remained the same as before. Even though overall results improved for LDA after normalization, the improvement was smallest as compared to the other classifiers with only an increase of 4% in overall accuracy. Results for LDA remained very similar to that of the SVM.

	Fall	Walk	Stand	Sit
Fall	100	0	0	0
Walk	0	100	0	0
Stand	0	0	100	0
Sit	34	0	8	58

Figure 4.11: Confusion matrix for LDA with normalized data

- ***Naïve Bayes:*** Results for NB also improved from 90% to 96% after normalization. Recall for Fall and Walk improved to 100%, precision also improved. In this case, 17% of the sit videos were classified as Fall decreasing the recall for Sit and precision for Fall.

	Fall	Walk	Stand	Sit
Fall	100	0	0	0
Walk	0	100	0	0
Stand	0	0	100	0
Sit	17	0	0	83

Figure 4.12: Confusion matrix for NB with normalized data

- ***BPNN:*** The results for BPNN also improved after normalization. Recall and precision for Sit was 100%. Recall for Fall also improved remarkably from 92% to 98%. Lowest recall was for Stand with many stand samples wrongly identified as Walk hence decreasing the precision for Walk.

	Fall	Walk	Stand	Sit
Fall	98	1	1	0
Walk	1	96	3	0
Stand	0	10	90	0
Sit	0	0	0	100

Figure 4.13: Confusion matrix for BPNN with normalized data

Figure 4.14 presents a comparison of accuracies for all the classifiers.

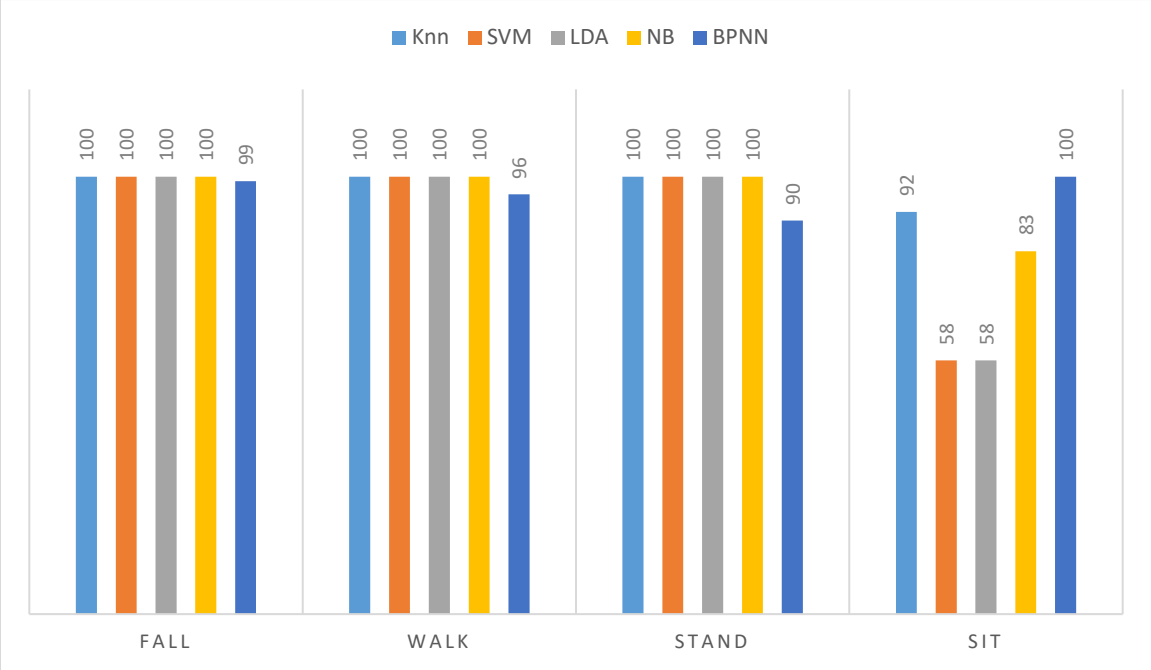


Figure 4.14: Comparison of accuracy with normalized data

4.6 Comparison with State of the Art:

Since the algorithm was developed by combining two separate datasets with different activities, only recall values are compared for individual classes with the state of the art. Poonsri et al (2017) used 58 video sequences of fall and no fall from Le2i dataset. They have used Mixture of Gaussian models for background subtraction. Then they extracted features like aspect ratio, area ratio and orientation of the human silhouette using PCA [43]. Marcos et al (2017) used convolutional neural networks to identify falls in video sequences. They report their results on

complete Le2i dataset for two classes i.e. ‘fall’ and ‘no fall’ [44]. Results of Fall detection are compared with the proposed method in Table 4-4.

Table 4-4: Comparison of Recall for class Fall with the state of the art

Proposed Method (with normalized data)						**Poonsri et al (2017)**	**Marcos et al (2017)**
Technique	**KNN**	**SVM**	**LDA**	**NB**	**BPNN**	**MoG, PCA**	**Optical flow, CNN**
	100%	100%	100%	100%	99%	93%	99%

Hbali et al (2017) use 3D skeleton based features for activity recognition. They use joint positions to calculate the Minkowski and Cosine distances between the joints to extract spatial features. Then for temporal features, they computed difference between the coordinates of each joint in the video sequence. For learning human actions, they have used random forest algorithm. They have reported their results on MSR Daily activity dataset [45]. Tamou et al (2016) have used 3D positions of the skeletal joints for activity recognition. They calculate difference between positions of the joints to obtain posture information in the current frame. Then they compute difference between joints’ position in the current frame and the initial frame to obtain temporal information. After getting spatial and temporal information in each frame, they calculate a mean feature vector to represent the video sequence. This feature vector is then input to the random forest classifier for action recognition [46]. Jalal et al (2017) have used human silhouette information along with joints’ position in the skeleton for activity recognition. The silhouettes and joint positions are obtained directly from RGB-D camera. First they apply pixel differentiation method for background noise removal. They obtain depth intensity centre values using connected component labelling, then they monitor neighbouring pixels intensity values in the consecutive frames to remove objects like doors etc. and to obtain human silhouettes. Then they extract multiple features from these silhouettes including joints’ angles and distances within a frame and frame wise joints’ positions differences, pixel intensities and gradient orientation. For motion detection, they have used difference of coordinates between consecutive frames. They have used Hidden Markov Models (HMMs) for training and testing purpose [47]. The results for walk and sit are compared in table 4-5.

Table 4-5: Comparison of Recall for classes Walk and Sit with the state of the art

Proposed Method (with normalized data)						Hbali et al (2017)	Tamou et al (2016)	Jalal et al (2017)
Technique / Class	**KNN**	**SVM**	**LDA**	**NB**	**BPNN**	**skeleton & depth features, RF**	**skeleton & depth features, RF**	**skeleton & depth features, HMMs**
Walk	100%	100%	100%	100%	96%	88%	95%	96%
Sit	92%	58%	58%	83%	100%	75%	90%	87%

Wang et al (2016) prepared their own dataset using Kinect device to recognize five potures including standing, kneeling, sitting, lying and stooping. They used background subtraction to remove the background and then extracted the human silhouette using connected component method. Then they calculated the centre of gravity of the human silhouette and divided the body into upper and lower halves. Then they calculated the horizontal projection histogram of the body region and used it to find the width of upper and lower body. Then various features were extracted including ratio of upper and lower body and distance between centre and edge contour. By plotting the distance of each contour pixel and centre of gravity, they obtained peak points which represent the tip of head, toes and hands. In this way a stick representation of the skeleton was extracted. They used LVQ neural network for activity classification [48]. Their results for standing posture are compared in table 4-6.

Table 4-6: Comparison of Recall for class Stand with the state of the art

Proposed Method (with normalized data)						Wang et al (2016)
Technique	**KNN**	**SVM**	**LDA**	**NB**	**BPNN**	**Silhouette & depth features, LVQ**
	100%	100%	100%	100%	90%	99%

It can be seen from the comparison results that proposed method gives quite remarkable performance for all the activities. All the classifiers performed well for fall, walk and stand classes. Sit was best detected using BPNN even though Knn also performed well. As a whole, Knn classifier gave best performance with an overall accuracy of 98% with normalized data.

CHAPTER 5: CONCLUSIONS AND FUTURE WORK

This thesis gives a comprehensive summary of the work carried out in the area of activity recognition using machine vision and machine learning techniques in recent years. It explores various methods used for features extraction and action recognition in the literature and discusses several applications in which activity recognition is being used.

The proposed methodology makes use of the 2D human skeleton position to extract useful features for activity recognition in the videos. Four activity classes are identified using five widely used supervised machine learning techniques and the results are compared to identify best suited technique for action classification using human skeleton. Various techniques have been used to extract useful features for action recognition in the literature. Usually, skeleton data is used in combination with the depth information which limits the scope of the research to specific type of cameras like Kinect. This study shows that promising results can be achieved by using only skeleton data from 2D video streams without any requirement of the depth information. Comparison of the results with the state of the art show that with some further work, this approach can be utilized in implementing the vision based activity recognition in surveillance and monitoring applications using standard CCTV cameras.

This study provides basis for further research in this area. This work can be extended to detect and classify more activities involving multiple person interactions which can lead to a better understanding of human behaviour in an integrated environment. Human behaviour understanding is a very important step in video surveillance applications to prevent crimes and timely action in case of an emergency. This work can also be extended to include more features along with 2D skeleton positions to extract more useful information from the scene like object detection can help in understanding the human to object interactions. This can help largely in ambient assisted living for elderly, in smart environments and robotics.

APPENDIX A

Activity Recognition Code

1) Code for calculating Train and Test Feature Vectors:

```
clear;
clc;
av_frames=10;                                   % set number of frames for averaging in post processing step
folderpath='G:\thesis\test data features -openpose\fall2';
cd(folderpath);
tfiles=dir('*.yml');                            %read openpose output files (frame wise) from the directory
numfiles=length(tfiles);
mydata=cell(1,numfiles);
for k=1:numfiles
   mydata{k}=cv.FileStorage(tfiles(k).name);    % store the pose information
end
p=zeros(1,numfiles);
for k=1:numfiles
   [p(k),~,~]=size(mydata{k}.pose_0);           % find number of persons in all frames
end
-----------------------------------------------------------------------------------------------------------------------
pixnum=12;
flag=0;
for i=1:numfiles
   if i<5 && p(i)==1                            % check the key points' values of actor and hold the values in an array
      frame_holdval=mydata{i}.pose_0(1,:,:);
      framenum_holdval=i;
      break;
   end
   frame_holdval=mydata{1}.pose_0(1,:,:);
   framenum_holdval=1;
end
strt=framenum_holdval+1;
for i=strt:numfiles
   if p(i)>1 && flag ==0
      value=mydata{i}.pose_0(:,:,:);
      out=bsxfun(@minus,frame_holdval,value);   %compare the values in current frame with the consecutive frame
      out=abs(out);
      dec=out(:,:,1:2)<pixnum;                  %check if the values are greater than a threshold
      for j=1:p(i)
         dec2(j)=nnz(dec(j,:,:)==1);
      end
      [~,I]=max(dec2);
      frame_holdval=mydata{i}.pose_0(I,:,:);    %recover values of the keypoints of the actor
      mydata{i}.pose_0(1,:,:)=frame_holdval;
   elseif p(i)==1
      frame_holdval=mydata{i}.pose_0(1,:,:);
      flag=0;
   elseif p(i)==0
      flag=1;
   elseif  p(i)>1 && flag ==1
```

```
        frame_holdval=mydata{i}.pose_0(1,:,:);
        flag=0;
    end
    dec2=0;
end
-------------------------------------------------End Preprocessing-------------------------------------------------
r_sh=zeros(numfiles,2);     %initialize arrays for obtaining joint positions for shoulder, hip, knee, ankle
l_sh=zeros(numfiles,2);
r_hp=zeros(numfiles,2);
l_hp=zeros(numfiles,2);
r_ne=zeros(numfiles,2);
l_ne=zeros(numfiles,2);
r_ank=zeros(numfiles,2);
l_ank=zeros(numfiles,2);
%-----obtain joint positions in separate arrays for all the frames in a video, assign 0 for empty frames------
for k=1:numfiles
    if(isempty(mydata{k}.pose_0))
        r_sh(k,:)=[0 0];
        l_sh(k,:)=[0 0];
        r_hp(k,:)=[0 0];
        l_hp(k,:)=[0 0];
        r_ne(k,:)=[0 0];
        l_ne(k,:)=[0 0];
        r_ank(k,:)=[0 0];
        l_ank(k,:)=[0 0];
    else
        r_sh(k,:)=[mydata{k}.pose_0(1,3,1) mydata{k}.pose_0(1,3,2)];
        l_sh(k,:)=[mydata{k}.pose_0(1,6,1) mydata{k}.pose_0(1,6,2)];
        r_hp(k,:)=[mydata{k}.pose_0(1,9,1) mydata{k}.pose_0(1,9,2)];
        l_hp(k,:)=[mydata{k}.pose_0(1,12,1) mydata{k}.pose_0(1,12,2)];
        r_ne(k,:)=[mydata{k}.pose_0(1,10,1) mydata{k}.pose_0(1,10,2)];
        l_ne(k,:)=[mydata{k}.pose_0(1,13,1) mydata{k}.pose_0(1,13,2)];
        r_ank(k,:)=[mydata{k}.pose_0(1,11,1) mydata{k}.pose_0(1,11,2)];
        l_ank(k,:)=[mydata{k}.pose_0(1,14,1) mydata{k}.pose_0(1,14,2)];
    end
end
%-------------------------------------------------------------------------------------------------------------------
%-----------------------------------------calculate joint angles and distances-----------------------------------------
    rd=zeros(1,numfiles);
    ld=zeros(1,numfiles);
    rd_hpnee=zeros(1,numfiles);
    r_hpnee=zeros(1,numfiles);
    ld_hpnee=zeros(1,numfiles);
    l_hpnee=zeros(1,numfiles);
    r_hpank=zeros(1,numfiles);
    l_hpank=zeros(1,numfiles);
    rd_neank=zeros(1,numfiles);
    r_neank=zeros(1,numfiles);
    ld_neank=zeros(1,numfiles);
    l_neank=zeros(1,numfiles);
for k=1:numfiles
  if k~=1
      if r_hp(k,:)== 0
          r_hp(k,:) = r_hp(k-1,:);
      end
```

```
        if l_hp(k,:)== 0
            l_hp(k,:) = l_hp(k-1,:);
        end
        if r_ne(k,:)== 0
            r_ne(k,:) = r_ne(k-1,:);
        end
        if l_ne(k,:)== 0
            l_ne(k,:) = l_ne(k-1,:);
        end
        if r_ank(k,:)== 0
            r_ank(k,:) = r_ank(k-1,:);
        end
        if l_ank(k,:)== 0
            l_ank(k,:) = l_ank(k-1,:);
        end
        if r_sh(k,:)== 0
            r_sh(k,:) = r_sh(k-1,:);
        end
        if l_sh(k,:)== 0
            l_sh(k,:) = l_sh(k-1,:);
        end
    end
        [rd_hpnee(k),r_hpnee(k)]=distance(r_hp(k,:),r_ne(k,:));
        [ld_hpnee(k),l_hpnee(k)]=distance(l_hp(k,:),l_ne(k,:));
        [~,r_hpank(k)]=distance(r_hp(k,:),r_ank(k,:));
        [~,l_hpank(k)]=distance(l_hp(k,:),l_ank(k,:));
        [rd_neank(k),r_neank(k)]=distance(r_ne(k,:),r_ank(k,:));
        [ld_neank(k),l_neank(k)]=distance(l_ne(k,:),l_ank(k,:));
 end
%-------------------------------------------------------------------------------------------------------------------------
%-----------------------------------------------calculate displacements-------------------------------------------------
[disp_rsh,disp_lsh,disp_rhp,disp_lhp,...
    disp_rne,disp_lne,disp_rank,disp_lank]=deal(zeros(1,numfiles));
disp_rsh(1)=0;
disp_lsh(1)=0;
disp_rhp(1)=0;
disp_lhp(1)=0;
disp_rne(1)=0;
disp_lne(1)=0;
disp_rank(1)=0;
disp_lank(1)=0;
for k=2:numfiles
    [disp_rsh(k),~]=distance(r_sh(k,:),r_sh(k-1,:));
    [disp_lsh(k),~]=distance(l_sh(k,:),l_sh(k-1,:));
    [disp_rhp(k),~]=distance(r_hp(k,:),r_hp(k-1,:));
    [disp_lhp(k),~]=distance(l_hp(k,:),l_hp(k-1,:));
    [disp_rne(k),~]=distance(r_ne(k,:),r_ne(k-1,:));
    [disp_lne(k),~]=distance(l_ne(k,:),l_ne(k-1,:));
    [disp_rank(k),~]=distance(r_ank(k,:),r_ank(k-1,:));
    [disp_lank(k),~]=distance(l_ank(k,:),l_ank(k-1,:));
end
-------------------------------------------------------------------------------------------------------------------------
feature_t=zeros(numfiles,16);
for k=1:numfiles
    rd(k)=rd_hpnee(k)/rd_neank(k);                    %ratio of distances between hip, knee and knee ankle
    ld(k)=ld_hpnee(k)/ld_neank(k);
```

```
end
  for k=1:numfiles
     feature_t(k,:)=[r_hpnee(k),l_hpnee(k),r_hpank(k),l_hpank(k),...        %feature vector before averaging
        r_neank(k),l_neank(k),rd(k),ld(k),disp_rsh(k),disp_lsh(k),...
        disp_rhp(k),disp_lhp(k),disp_rne(k),disp_lne(k),...
        disp_rank(k),disp_lank(k)];
  end

   feature_t =feature_t';
%---------------------------------------------------- Post Processing----------------------------------------------------
   temp=round(numfiles/av_frames);
feature= zeros(16,temp);
for s=1:16                                  % Calculate average of features in a window defined by av_frames
   m=1;
   j=1;
for i=1:av_frames:numfiles
   sums=feature_t(s,i);
   for k=1:av_frames
   if mod(m,av_frames) ~= 0
      sums = sums +feature_t(s,m+1);
      m = m +1;
   end
   end
   av=sums/av_frames;
   feature(s,j)=av;
   j=j+1;
   m=m+1;
end
end
feature=feature';                           %final feature vector

%--------------------------------------function for calculating angles ad distances--------------------------------
function [dist,theta]=distance( a,  b)
   dy = b(2) - a(2);
   dx = b(1) - a(1);
   theta = atan2d(dy, dx);
   dist=sqrt(dy^2 +dx^2);
end
%-----------------------------------------------------------------------------------------------------------------------
```

2) Code for classification using Back Propagation Neural Network:

```
clear;
clc;
d=xlsread('G:\thesis\train_norm_bp.xlsx');          % Path of train vector
l=xlsread('G:\thesis\label_bp.xlsx');               % Path of true labels
nprtool;
{Implemented using 14 hidden layer neurons and 35% test data, 15% validation and 50% train data}
%-----------------------------------------------------------------------------------------------------------------------
```

3) Code for classification using KNN, SVM, LDA, Naïve Bayes classifiers:

```
clear;
clc;
trainfeat=xlsread('G:\thesis\train data features\train_norm.xlsx');     %path of train vector
label=xlsread('G:\thesis\train data features \label.xlsx');             %path of labels
cd('G:\thesis\test data features norm');                                 %path of test feature vectors
X = trainfeat;
Y = label;
```

```
con=1;
flag=1;
while con
   test= input('Enter the name of test file\n','s');                    %user inputs the name of test feature vector
   testfeat=strcat(test,'.xlsx');
   Z=xlsread(testfeat);
   if(flag==1)
   disp('Choose Classifier: Press');                  %choose between four of the classifiers used for classification
   classifier=input(' k for Knn\n s for SVM\n l for LDA\n n for Naive Bayes\n','s');
   if(classifier == 'k')
      md1 =fitcknn(X,Y,'NumNeighbors',4,'Distance','mahalanobis');     %calculate model for knn classifier
      flag=0;
   elseif(classifier == 's')
      md1 = fitcecoc(X,Y);                                             %calculate model for SVM classifier
      flag=0;
   elseif(classifier == 'l')
      md1 = fitcdiscr(X,Y);                                %calculate model for linear discriminant classifier
      flag=0;
   elseif(classifier == 'n')
      md1 = fitcnb(X,Y);                                   %calculate model for Naïve Bayes classifier
      flag=0;
   else
      disp('You entered wrong character');
      disp('Please try again');
      flag=1;
   end
   end
   if(flag==0)
      Type = predict(md1,Z);                               %predict type of test video based on the model
      if (Type == 1)
         disp('faint');
         msgbox('faint');
      elseif (Type == 2)
         disp('walk');
         msgbox('walk');
      elseif (Type == 3)
         disp('stand');
         msgbox('stand');
      elseif (Type == 4)
         disp('sit');
         msgbox('sit');
      else
          val=unique(Type);
          instances=hist(Type,val);
          [M,I]=max(instances);
          type=val(I);
          if (type == 1)
            disp('faint');
            msgbox('faint');
          elseif (type == 2)
            disp('walk');
            msgbox('walk');
          elseif (type == 3)
            disp('stand');
            msgbox('stand');
          elseif (type == 4)
```

```
            disp('sit');
            msgbox('sit');
        end
    end
  disp('Do you wish to continue? y\n');
  d=input('','s');
  if d=='y' || d=='Y'
    con=1;
    flag=1;
    clc;
  else con=0;
  end
  end
end
```

REFERENCES

[1] Jainy, M.; Gemerty, J. C.; and Snoek, C. G. M.; "What do 15,000 object categories tell us about classifying and localizing actions?"; *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Boston, MA)*, 46–55, 2015

[2] Wrzalik, M.; Krechel, D.; "Human Action Recognition Using Optical Flow and Convolutional Neural Networks"; *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Dec 2017.

[3] Zhang, Z.;Wang, C. Xiao, B.;Zhou, W.; Liu, S.; "Robust relative attributes for human action recognition"; *Pattern Analysis and Applications*, vol. 18, Issue 1, pp 157–171,Feb 2015.

[4] Zhu, G.; Zhang, L.; Shen, P.; Song, J.; "An Online Continuous Human Action Recognition Algorithm Based on the Kinect Sensor"; *Sensors*, Jan 2016.

[5] Song, Y.; Morency, L. P.; Davis, R.; "Action recognition by hierarchical sequence summarization"; *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Portland, OR)*, 3562–3569, 2013

[6] Yang, X.; Tian, Y.; "Super normal vector for human activity recognition with depth cameras," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, pp. 1028–1039, 2017.

[7] Wu, Q.; Wang, Z.; Deng, F.; Chi, Z.; and Feng, D. D.; "Realistic human action recognition with multimodal feature selection and fusion"; *IEEE Trans. Syst. Man Cybern. Syst. 43*, 875–885, 2013.

[8] Zerrouki, N.; Houacine, A.; "Automatic Classification of Human Body Postures Based on the Truncated SVD" *Journal of Advances in Computer Networks*, vol. 2, no. 1, March 2014.

[9] Yang, Y.; Saleemi, I.; Shah, M.; "Discovering Motion Primitives for Unsupervised Grouping and One-shot Learning of Human Actions, Gestures, and Expressions"; *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35, 1635–1648, 2013.

[10] Li, C.; Hua, T.; "Human Action Recognition Based on Template Matching"; *Procedia Engineering*, vol. 15, 2824-2830, 2011.

[11] Bianco, S.; Buzzelli, M.; Schettini, R.; "Object Detection Using Feature-based Template Matching"; *Proceedings of SPIE - The International Society for Optical Engineering*, 8661:86610C, March 2013.

[12] Araujo, A.; Girod, B.; "Large-Scale Video Retrieval Using Image Queries"; *IEEE Transactions on Circuits and Systems for video technology*, vol. 28, no. 6, June 2018.

[13] Olatunji, I.E.; "Human Activity Recognition for Mobile Robot"; *Journal of Physics: Conference Series*, vol. 1069, conference 1, Jan 2018.

[14] Barriga, A.;Conejero, J.M.; Harnandez, J.; Jurado, E.; Moguel, E.;Figueroa, F.; "A Vision-Based Approach for Building Telecare and Telerehabilitation Services"; *Sensors*, vol. 16, no. 10, Oct 2016.

[15] Lee, F.; "Ambient Intelligence—The Ultimate IoT Use Cases"; Oct 2017, [Online] Available: https://medium.com/iotforall/ambient-intelligence-the-ultimate-iot-use-cases-5e854485e1e7

[16] Chong, Y.S.; Tay, Y.H.; "Abnormal Event Detection in Videos using Spatiotemporal Autoencoder"; *Advances in Neural Networks - ISNN 2017*, Lecture Notes in Computer Science, vol 10262. Springer, Jan 2017.

[17] Manzi, A.; Dario, P.; Cavallo F. ;. "A human activity recognition system based on dynamic clustering of Skeleton data"; *Sensors (Basel)*, May 2017

[18] Le, T.; Nguyen, M.; Nguyen, T.; "Human Posture recognition using human skeleton provided by Kinect"; *2013 International Conference on Computing, Management and Telecommunications* , Jan 2013.

[19] Cippitell, E.; Gasparrini, S.; Gambi, E.; Spinsante, S.; "A Human Activity Recognition System Using Skeleton Data from RGBD Sensors"; *Computational Intelligence and Neuroscience*, vol. 2016, Article ID 4351435, 14 pages, 2016.

[20] Li, M.; Leung, H.; Shum, H.P.H.; "Human action recognition via skeletal and depth based feature fusion"; *Proceedings of the 9th International Conference on Motion in Games*, 124-132, Oct 2016.

[21] Kushwaha, A.K.S.; Srivastava, M.R.; "A framework for human activity recognition using pose feature for video surveillance system"; *International Journal of Computer Applications*, 0975 – 8887,2016.

[22] Zerrouki,N.; Harrou, F.; Sun, Y.; Houacine, A.; "Vision-based Human Action Classification Using Adaptive Boosting Algorithm"; *IEEE Sensors Journal*, May 2018.

[23] Goudelis, G.; Tsatiris, G.; Karpouzis, K.; "Fall detection using History Triple Features"; *Proceedings of the 8th ACM International Conference on Pervasive Technologies Related to Assistive Environments*, no. 81, July 2015

[24] Mu, C.; Xie, J.;Yan, W.; Liu, T.; Li, P.; "A fast recognition algorithm for suspicious behaviour in high definition videos"; *Multimedia Systems*, vol. 22, Issue 3, pp 275–285, June 2016.

[25] Xia, L.; Yang, B.; Tu, H.; "Recognition of suspicious behaviour using case based reasoning"; *Journal of Central South University*, vol. 22, Issue 1, pp 241–250, Jan 2015.

[26] Wachs, J.P.; Kölsch, M.; Goshorn, D.; "Human Posture recognition for intelligent vehicles"; *Journal of Real-Time Image Processing*, vol 5, Issue 4, pp 231–244, Dec 2010.

[27] Wang. L.; Qiao, Y.; Tang, X.; "Action recognition and detection by combining motion and appearance features", *THUMOS14 Action Recognition Challenge*, 1-6, 2014.

[28] Albawendi, S.; Lotfi, A.; Powell, H.; Appiah, K.; "Video Based Fall Detection using Features of Motion, Shape and Histogram"; *Proceedings of the 11th Pervasive Technologies Related to Assistive Environments Conference*, June 2018.

[29] Openpose library, [Online] Available: https://github.com/CMU-Perceptual-Computing-Lab/openpose

[30] Cao, Z.; Simon, T.; Wei, S.; Sheikh, Y.; "Real time Multi Person 2D Pose Estimation using Part Affinity Fields"; *CVPR*, 2017.

[31] Maladkar, K.; "Types Of Activation Functions In Neural Networks And Rationale Behind It"; 2018, [Online] Available: https://www.analyticsindiamag.com/most-common-activation-functions-in-neural-networks-and-rationale-behind-it/

[32] Goodfellow, I.; Bengio, Y.; Courville, A.; *Deep Learning*; MIT Press, 2016, [Online] Available: http://www.deeplearningbook.org

[33] S. N. Sivanandam, S. Sumathi, S. N. Deepa; *Introduction to neural networks using Matlab 6.0 Computer engineering series*; New Delhi: Tata McGraw-Hill, 2006

[34] Gurney K; *An Introduction to neural networks;* London: UCL Press Limited, 1997.

[35] Bronshtein, A.; "A Quick Introduction to K-Nearest Neighbors Algorithm"; 2017, [Online] Available: https://medium.com/@adi.bronshtein/a-quick-introduction-to-k-nearest-neighbors-algorithm-62214cea29c7

[36] Duda, R. O.; Hart, P. E.;Stork, D.G.; *Pattern Classification*; 2nd Ed; New York: Wiley-Interscience, 2000

[37] McCormick, C.; "Mahalanobis distance"; 2014, [Online] Available: http://mccormickml.com/2014/07/22/mahalanobis-distance/

[38] Burges, C. J. C.; "A Tutorial on Support Vector Machines for Pattern Recognition"; 1998, [Online] Available: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/svmtutorial.pdf

[39] Ben-Hur, A.; Weston, J.; "A User's Guide to Support Vector Machines"; [Online] Available: http://pyml.sourceforge.net/doc/howto.pdf

[40] Alpaydin, E.; *Introduction to Machine Learning;* London: The MIT Press, 2nd Edition 2010.

[41] Li, W.; *MSR Daily Activity 3D Dataset*; 2012; [Online] Available: https://www.uow.edu.au/~wanqing/#Datasets

[42] *Le2i Fall Detection Dataset;* 2013; [Online] Available: http://le2i.cnrs.fr/Fall-detection-Dataset?lang=fr

[43] Poonsri, A.; Chiracharit, W.; "Fall Detection Using Gaussian Mixture Model and Principle Component Analysis"; *9th International Conference on Information Technology and Electrical Engineering (ICITEE), Phuket, Thailand*, 2017

[44] Núñez-Marcos, A.; Azkune, G.; Arganda-Carreras, I.; "Vision-Based Fall Detection with Convolutional Neural Networks"; *Wireless Communications and Mobile Computing*, Dec 2017.

[45] Hbali, Y.; Hbali, S.; Lahoucine, B.; Mohammed, S.; "Skeleton-based human activity recognition for elderly monitoring systems"; *IET Computer Vision 12(1),* August 2017.

[46] Tamou, A.; Ballihi, L., Aboutajdine, D.; "Automatic Learning of Articulated Skeletons based on Mean of 3D Joints for Efficient Action Recognition"; *International Journal of Pattern Recognition and Artificial Intelligence,* Sept 2016.

[47] Jalal, A.; Kamal, S.; Kim, D.; "A Depth Video-based Human Detection and Activity Recognition using Multi-features and Embedded Hidden Markov Models for Health Care Monitoring Systems"; *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 4, no. 4, Jan 2017

[48] Wang, W.; Chang, J.; Haung, S.; Wang, R.; "Human Posture Recognition Based on Images Captured by the Kinect Sensor"; *International Journal of Advanced Robotic Systems*, 2016

Completion Certificate

It is certified that the thesis titled **"Vision based Human Activity Recognition using Skeleton Data"** submitted by registration no. 00000118185, NS Sumaira Ghazal of MS-86 Mechatronics Engineering is completed in all respects as per the requirements of Main Office, NUST (Exam branch).

Supervisor: _____
Dr. Umar Shahbaz Khan
Date: ____ Jan, 2019