

Advanced Video Analytic System for Posture and Activity
Recognition: Leveraging MediaPipe, CNN-LSTM, and
Ensemble Learning for Fall and Unstable Motion Detection

by

Farhan Md. Siraj
22266023

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
M.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
October 2024

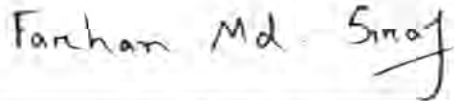
© 2024. Brac University
All rights reserved.

Declaration

It is hereby declared that

1. The thesis submitted is my own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. I have acknowledged all main sources of help.

Student's Full Name & Signature:



Farhan Md. Siraj
ID: 22266023

Approval

The thesis titled “Advanced Video Analytic System for Posture and Activity Recognition: Leveraging MediaPipe, CNN-LSTM, and Ensemble Learning for Fall and Unstable Motion Detection” submitted by

1. Farhan Md. Siraj (22266023)

Of Summer, 2024 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of M.Sc. in Computer Science on October 19, 2024.

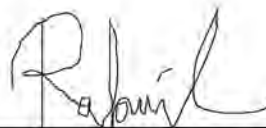
Examining Committee:

External Examiner:
(Member)



Dr. Taskeed Jabid
Professor
Department of Computer Science and Engineering
East West University

Internal Examiner:
(Member)



Dr. Md. Golam Rabiul Alam
Professor
Department of Computer Science and Engineering
BRAC University

Supervisor:
(Member)



Dr. Aniqua Nusrat Zereen
Assistant Professor
Department of Computer Science and Engineering
BRAC University

Program Coordinator:
(Member)

A handwritten signature in black ink, enclosed in an oval shape. The signature appears to be "Sadek".

Dr. Md Sadek Ferdous
Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

A handwritten signature in black ink, enclosed in a horizontal line. The signature appears to be "Sadia Hamid Kazi".

Sadia Hamid Kazi, Ph.D.
Chairperson and Associate Professor
Department of Computer Science and Engineering
BRAC University

Ethics Statement

The thesis was carried out in strict compliance with BRAC University's research ethics, norms, and codes of practice. All referenced sources have been appropriately cited. As the sole author, I take full responsibility for any ethical violations that may occur. The research was conducted with careful attention to ethical standards, reflecting a strong commitment to adhering to the guidelines established by BRAC University throughout the study.

Abstract

Human posture detection and classification are vital in monitoring activities, especially in health and safety contexts, such as fall detection in elderly care. This thesis presents a comparative study of two machine learning approaches for real-time human posture classification using real time video data, a traditional feature-based approach using a Voting Classifier and a deep learning approach utilizing a Convolutional Neural Network-Long Short-Term Memory (CNN-LSTM) model. The feature-based method incorporates pose estimation using MediaPipe to extract human body landmarks, followed by classification using an ensemble of Random Forest, Support Vector Machine (SVM), and K-Nearest Neighbors (KNN). On the other hand, the CNN-LSTM model captures both spatial and temporal dynamics of video sequences by extracting visual features through Convolutional Neural Network(CNN) and modelling temporal dependencies via LSTM. The models are evaluated on a dataset of four postures— Fall, Sit, Stand, and Unstable—with promising results. This work demonstrates the effectiveness of combining pose-based features with voting classifiers and the power of deep learning in sequential data, offering a robust solution for real-time posture classification systems.

Keywords: Posture Detection; Human Activity Recognition; Fall Detection; Voting Classifier; CNN-LSTM; Machine Learning; Random Forest; SVM; KNN, Pose Estimation; MediaPipe.

Acknowledgement

I am Farhan Md. Siraj and I attend BRAC University's Computer Science and Engineering (CSE) department. This work is my final thesis and the result of two years of scholarly investigation. My supervisor, Dr. Aniqua Nusrat Zereen, has provided me with tremendous guidance, constant support, and intellectual insights throughout this research, for which I am incredibly grateful. Her commitment to quality work and creating a stimulating learning atmosphere has dramatically influenced our study. I appreciate Dr. Aniqua's tolerance, support, and the several hours she spent as my mentor. Her wisdom and astute advice have inspired me to seek information more actively. I would also like to thank the Asian Institute of Technology, Thailand, for providing the dataset used in this research. I also want to express my deepest gratitude to my parents for their unwavering support, faith in my abilities, and love. Their selflessness, support, and steadfast determination have fuelled this scholastic journey. Their unselfish dedication to my education and well-being has been an enormous source of inspiration and strength. Finally, I would like to thank the Almighty Allah, whose blessings allowed me to finish my thesis without facing many significant challenges.

Table of Contents

Declaration	i
Approval	ii
Ethics Statement	iv
Abstract	v
Acknowledgment	vi
Table of Contents	vii
List of Figures	ix
List of Tables	x
Nomenclature	x
1 Introduction	1
2 Related Work	3
3 Methodology	7
3.1 Dataset Preparation	10
3.1.1 Dataset Preparation for Traditional Machine Learning Classifiers	10
3.1.2 Dataset Preparation for CNN-LSTM	10
3.2 Feature Extraction	12
3.2.1 Landmark Detection	12
3.3 Traditional Machine Learning Classifiers	15
3.3.1 Random Forest Classifier	15
3.3.2 Support Vector Machine (SVM)	15
3.3.3 K-Nearest Neighbors (KNN)	15
3.4 Voting Classifier	15
3.5 Deep Learning with CNN-LSTM	16
3.5.1 Feature Extraction for CNN	16
3.5.2 Model Architecture	16
3.6 Enhanced Features	18
3.6.1 Fall Detection Using Knee Angle	18
3.6.2 Angle and Distance Calculation	19

3.7	Implementation Aspects	20
3.7.1	Libraries and Tools	20
3.7.2	Hyperparameter Tuning	21
3.7.3	Model Training and Parameters	21
3.7.4	Evaluation Metrics	21
4	Results and Performance Analysis	23
4.0.1	Comparative Analysis of Model Performance	23
4.0.2	Confusion Matrix	23
4.0.3	ROC Curves	25
4.0.4	Classification Results	27
4.0.5	Visual Representation of Classified Activities	28
5	Conclusion	31
	Bibliography	36

List of Figures

3.1	Data collection phase of the system.	8
3.2	Data preparation and training phases of the system.	9
3.3	Evaluation results showing predicted and actual labels for the four activities.	9
3.4	Snapshots of Activity Classes from the Dataset	12
3.5	Visualization of MediaPipe Pose Landmarks	13
3.6	Dataset Preparation: Snapshots of Activities after applying Mediapipe	14
3.7	CNN-LSTM architecture for human activity recognition.	17
3.8	Fall based on angle.	19
4.1	Normalized Confusion Matrix for Voting Classifier.	24
4.2	Normalized Confusion Matrix for CNN-LSTM Model.	25
4.3	ROC Curves for Voting Classifier.	26
4.4	ROC Curves for CNN-LSTM Model.	27
4.5	A fall incident forecasted by the trained model.	28
4.6	An unstable condition predicted by the trained model.	29
4.7	A standing posture predicted by the trained model.	29
4.8	A sitting posture forecasted by the trained model.	30

List of Tables

2.1	Comparison of Studies Utilizing MediaPipe	6
3.1	MediaPipe Pose Landmarks	13
4.1	Accuracy of Different Models	23
4.2	Classification Report for Voting Classifier	27
4.3	Classification Report for CNN-LSTM Model	28

Chapter 1

Introduction

Human posture recognition has become a critical area of research due to its wide-ranging applications in improving quality of life, particularly in elder care. Fall detection is a vital application where rapid intervention can significantly reduce injury severity and improve recovery outcomes. Falls are a leading cause of injury among the elderly, often resulting in long-term health complications. Studies, such as those by Al-Aama [1], reveal that over one-third of elderly individuals experience falls annually, with half suffering repeated incidents. Given the increasing elderly population, the demand for reliable and scalable fall detection systems has grown substantially. This research seeks to address these challenges by leveraging Artificial Intelligence (AI) advances, particularly video analytics and machine learning, to develop a nonintrusive system for posture recognition and fall detection.

Motivation

Monitoring elderly individuals in real-time is labour-intensive and costly, motivating the development of automated solutions. This research aims to address these challenges by leveraging advances in AI, particularly video analytics and machine learning, to develop a nonintrusive system for posture recognition and fall detection. Our proposed system operates in the background, offering real-time posture monitoring while maintaining the user's privacy and independence. Combining deep learning techniques with classical machine learning methods, we present a comprehensive approach capable of recognizing various human postures (e.g., sitting, standing, unstable movements) and accurately detecting falls. This work seeks to advance posture detection technology, contributing to a safer environment for the elderly and enhancing the quality of care provided.

Research Problem

Falls are a leading cause of injury among the elderly, often resulting in long-term health complications. Real-time monitoring is essential for early detection of falls and unsteady movements. Traditional monitoring systems, however, often intrude on privacy and independence or rely on wearable devices, which are not always feasible. This research addresses the need for an automated, non-intrusive posture recognition system to detect falls in real-time while maintaining a balance between accuracy and privacy.

Research Objectives

This research aims to develop a robust and efficient system that can analyze video data in real-time to classify human activities such as sitting, standing, falling, and unsteady movements. The system is particularly relevant for elderly care applications, where timely detection of falls can lead to life-saving interventions, underlining the critical importance of our work.

To achieve this, we propose a hybrid CNN-LSTM architecture, combined with classical ensemble models (Random Forest, SVM, KNN), to ensure both spatial and temporal analysis of human movement. MediaPipe is integrated into our system for accurate posture extraction from video streams, ensuring seamless real-time operation. The system is designed to handle diverse real-world conditions, ensuring high accuracy across various environments and scenarios.

Research Contributions

This paper makes the following key contributions:

1. **Hybrid Architecture:** We propose a novel hybrid architecture that combines CNN for spatial feature extraction and LSTM for temporal sequence modeling, enabling accurate classification of human postures and fall detection.
2. **Integration of Classical and Deep Learning Models:** In addition to the CNN-LSTM architecture, we implement an ensemble of traditional classifiers (Random Forest, SVM, KNN) to enhance robustness, demonstrating that a hybrid approach can outperform standalone models.
3. **Real-time Performance with MediaPipe:** The system integrates MediaPipe for real-time posture tracking and feature extraction, ensuring efficiency without compromising accuracy, making the system suitable for practical applications in elder care.

Thesis Organization

The remainder of this paper is organized as follows: Section 2 reviews related work on human posture recognition and fall detection systems, highlighting recent advances in deep learning, machine learning techniques, and MediaPipe. Section 3 details the proposed methodology, including the CNN-LSTM hybrid architecture, traditional classifiers with a voting classifier, and the integration of MediaPipe for real-time analysis. Section 4 presents an in-depth analysis of the results. Finally, we conclude with a discussion of the limitations and future directions in Section 5.

Chapter 2

Related Work

Delays in assisting following elders' falls lead to complications that can leave them with severe mobility issues. These have been the reasons for the wide adoption of wearable sensors in addressing the problem, including wristbands and fall-prevention shoes with embedded cameras and obstacle-detection lasers [2]. Such wearable devices inherently have significant drawbacks; however, they are susceptible to environmental noise and require a high level of compliance on the user's part, which may be unavailable if wearable devices are uncomfortable or forgotten [3].

Ali et al. [4] surveyed fall detection approaches, emphasizing the role of deep learning in improving detection accuracy. Their findings indicate that while deep learning techniques show promise, challenges remain in ensuring robustness across diverse environments and scenarios, particularly in low-light conditions.

Fixed optical sensors have also been explored in studies [5], ranging from human movement analyses to silhouette tracking over time [6] and extended studies of high-level human activities [7] [8]. Zhou et al. [6] demonstrated the effectiveness of extracting precise human silhouettes through foreground segmentation, morphological filtering, and continuous background updating techniques that help mitigate lighting and environmental fluctuations. Unlike silhouette extraction, our system uses the 2D skeleton joint vital points, which provide high-level spatial data, making the actions more recognizable. Features such as skeletons are also more discriminative than blobs of silhouettes since they give far more precise representations of human posture and movement.

Wang and Wu [9] developed a real-time fall detection system utilizing smartphone sensors. Their approach demonstrates the feasibility of using mobile devices for fall detection, emphasizing the importance of low-latency processing and high detection accuracy in critical applications such as elderly care.

Zhou et al. [10] proposed a zero-shot video object segmentation method that employs spatiotemporal object representations using an encoder-bridge-decoder structure. The two-stream encoder processes both input images and optical flow fields. In contrast, our system uses a single camera input stream and relies on skeleton-based pose estimation for action recognition. While depth cameras have been used for accurate spatial and temporal gait measurements [11], they are relatively large, expensive, and limited in range compared to standard RGB cameras.

Anguita et al. [12] introduced a public domain dataset designed for smartphone human activity recognition. This dataset includes recordings of various activities, enabling researchers to develop and evaluate models for recognizing human actions

from mobile sensor data. The work emphasizes the potential of utilizing everyday devices for accurate activity recognition.

Shahroudy et al. [13] presented the NTU RGB+D dataset, a large-scale benchmark for 3D human activity analysis, consisting of depth maps and RGB videos captured from various angles. This dataset enables the development of deep learning models for recognizing complex activities in 3D space, significantly advancing the state of the art in human activity recognition.

Wang et al. [14] proposed a concept of depth appearance-based local occupancy patterns, which model relationships between body parts and surrounding objects. The authors applied data mining techniques to identify the most helpful features for action classification on an action set of joints. Indeed, their system acquires sequences of depth maps and 3D joint positions from depth cameras, then extracts features based on the local occupancy patterns. In contrast, our system is straightforward. It takes advantage of multi-view RGB cameras and 2D skeleton joints during training.

Wu et al. [15] provided a comprehensive survey on human activity recognition in video, covering various methodologies, datasets, and evaluation metrics. Their analysis highlights the challenges and future directions in the field, including the need for more robust models that can generalize across different environments and scenarios.

Li et al. [16] developed a CNN-based skeleton pose detection system integrated with a transformer module for joint rearrangement, allowing the automatic selection of essential joints. Although their approach requires a max-out scheme to manage multiple individuals, our system is designed to handle multi-person scenarios in real time without such schemes. Zhang et al. [17] combined LSTM modules with hand-crafted and learned features to capture geometric relationships between joints and body segments in 3D skeletons. However, their approach suffers from overfitting due to many feature combinations. Our system avoids this by directly processing normalized 2D pose data without extending the feature space with hand-crafted descriptors.

Simonyan and Zisserman [18] proposed a two-stream convolutional network architecture for video action recognition, which utilizes both spatial and temporal information. Their approach demonstrated that combining features from static frames and motion can significantly improve action recognition performance in real-world scenarios.

Andrade et al. [19] presented a temporal CNN for learning spatiotemporal features on short video clips for human activity recognition. The architecture was based on 3D convolutional layers and convolutional LSTM units, similar to our RGB-based system. However, no real-time performances have been shown for this system.

Donahue et al. [20] introduced long-term recurrent convolutional networks (LRCNs) that CNNs with long short-term memory (LSTM) networks for visual recognition tasks. Their model effectively captures temporal dependencies, making it particularly suitable for human activity recognition in video sequences and improving accuracy.

Hussain et al. [21] proposed a lightweight deep learning model for real-time human activity recognition. Their work emphasizes the importance of designing efficient models that can run on resource-constrained devices while maintaining high accuracy, making them suitable for smart homes and healthcare monitoring applications.

It relies on MediaPipe Pose estimation [22] for real-time extraction of human pose landmarks from video frames. Features such as angles, distances, and ratios corresponding to different critical points of the body are extracted and require further processing. These features are then supplied to several machine learning classifiers, including Random Forest, Support Vector Machine, and K-Nearest Neighbors, combined into a Voting Classifier for robust classification. Additionally, the paper proposes a CNN-LSTM model that learns from sequences of video frames, capturing both spatial and temporal features. This significantly enhances the system's ability to recognize activities over time and boosts confidence in its performance. Table 2.1 summarizes some of the works on mediapipe.

This work combines these components to achieve detailed activity detection and posture categorization with classes such as Fall, Sit, Stand, and Unstable. The system will be analyzed using a confusion matrix analysis, plotting the ROC curve and a classification report to evaluate performance. The system is designed to be easily adaptable and deployable across various hardware platforms, making it highly scalable and applicable to real-world scenarios.

Table 2.1: Comparison of Studies Utilizing MediaPipe

Study	Approach	Limitations
Gupta et al. [23]	This study developed a real-time human pose recognition system using MediaPipe’s pose estimation capabilities integrated with deep learning frameworks, achieving notable accuracy.	The model faced challenges with occlusions and complex poses, reducing accuracy in dynamic environments.
Kumar et al. [24]	The authors proposed a hand gesture recognition system that leveraged MediaPipe’s hand tracking features, combined with machine learning classifiers for gesture detection.	The system struggled under varying lighting conditions and complex backgrounds, affecting recognition performance.
Othman et al. [25]	This research applied MediaPipe to extract pose landmarks for action recognition using a CNN, demonstrating competitive results in accuracy.	Limitations included difficulties in recognizing fast-paced actions and sensitivity to the quality of input videos.
Al-Sadi et al. [26]	A fitness monitoring system was developed that utilized MediaPipe for real-time analysis of motion and exercise techniques, providing feedback to users on their performance.	The system’s performance was hindered by crowded environments and varying camera angles, leading to inaccuracies in motion analysis.
Patel et al. [27]	This work utilized MediaPipe for real-time tracking of facial landmarks to improve human-computer interaction through emotion recognition.	Limitations included issues with accuracy in non-frontal facial orientations and environmental noise interference.
Chen et al. [28]	The authors presented a real-time activity recognition framework using MediaPipe for skeleton detection and classification, achieving high performance in recognizing daily activities.	The framework was sensitive to camera positioning and could struggle with overlapping activities.
Liu et al. [29]	This study implemented MediaPipe to create an interactive virtual try-on application for fashion, enabling users to visualize clothing on themselves using augmented reality.	Challenges included limitations in accurately tracking fast movements and ensuring stability in virtual overlays.
Cheng et al. [30]	The researchers developed a sign language recognition system employing MediaPipe for hand tracking, coupled with recurrent neural networks for gesture classification.	The system had trouble with recognizing signs in noisy backgrounds and complex lighting conditions, affecting accuracy.
Zhang et al. [31]	The study introduced a smart home automation system that utilized MediaPipe for person detection and gesture recognition to control devices.	Limitations included challenges with detecting gestures at a distance and in low-light conditions.
Jiang et al. [32]	This research focused on using MediaPipe to analyze human behavior in video surveillance, implementing action recognition to detect abnormal activities.	The approach faced limitations with distinguishing between similar actions and performing in diverse environmental conditions.

Chapter 3

Methodology

The proposed elder care video analytic system is designed to continuously monitor individuals in real-time using fixed cameras, classifying their activities to ensure their safety. The system detects four daily activities: sitting, standing, unstable movements, and falls. Unstable movements are considered critical, as they can precede or follow a fall caused by sudden dizziness, chest pain, tripping over obstacles, or accidental slips. The hardware setup includes IP cameras strategically placed to monitor areas where residents most likely require assistance.

Figure 3.1 illustrates the data collection phase of the system. This step involves recording and instructing volunteers to pose in the four activity classes (sitting, standing, unstable movements, and falls). Figure 3.2 shows both the data preparation and training phases, where features are extracted using MediaPipe, and classifiers are trained using traditional machine learning methods (KNN, SVM, RF) and a CNN-LSTM model. Finally, Figure 3.3 presents the evaluation results, displaying examples of the system's predictions for the four activity classes compared to the actual labels.

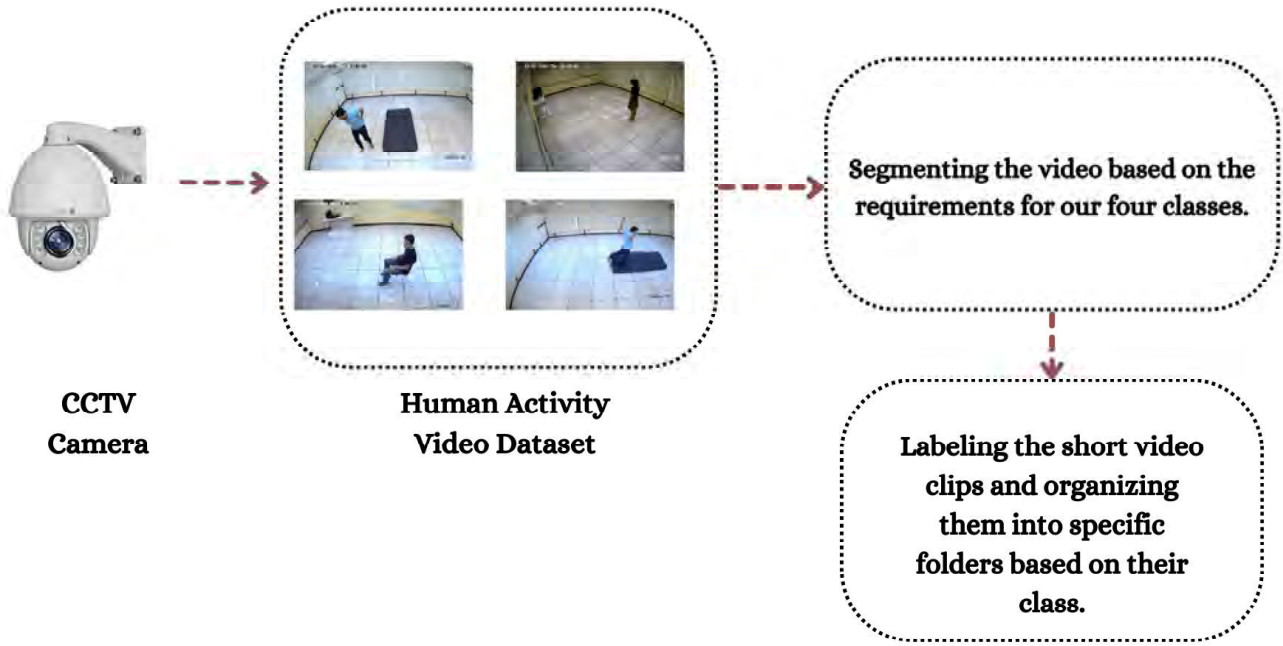


Figure 3.1: Data collection phase of the system.

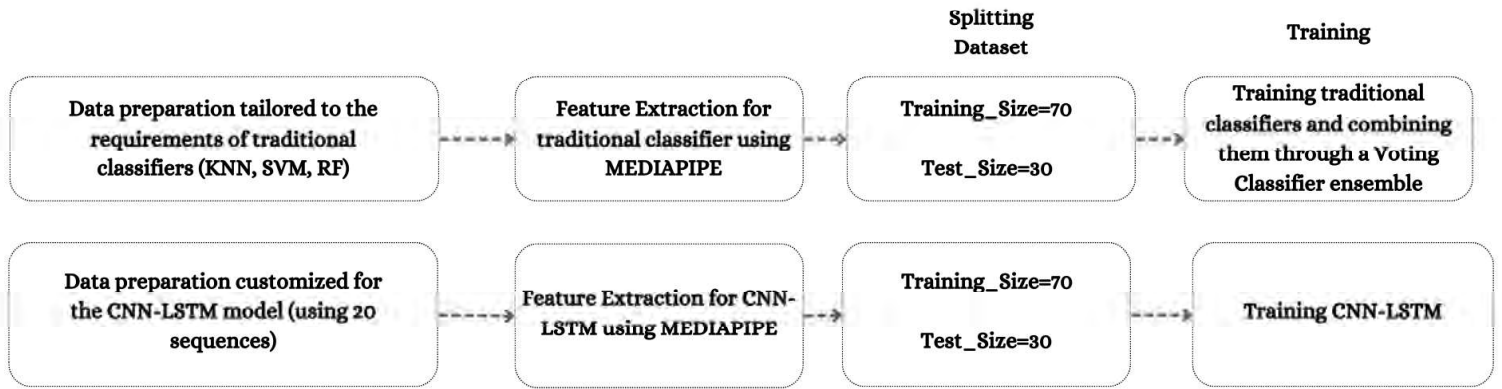


Figure 3.2: Data preparation and training phases of the system.

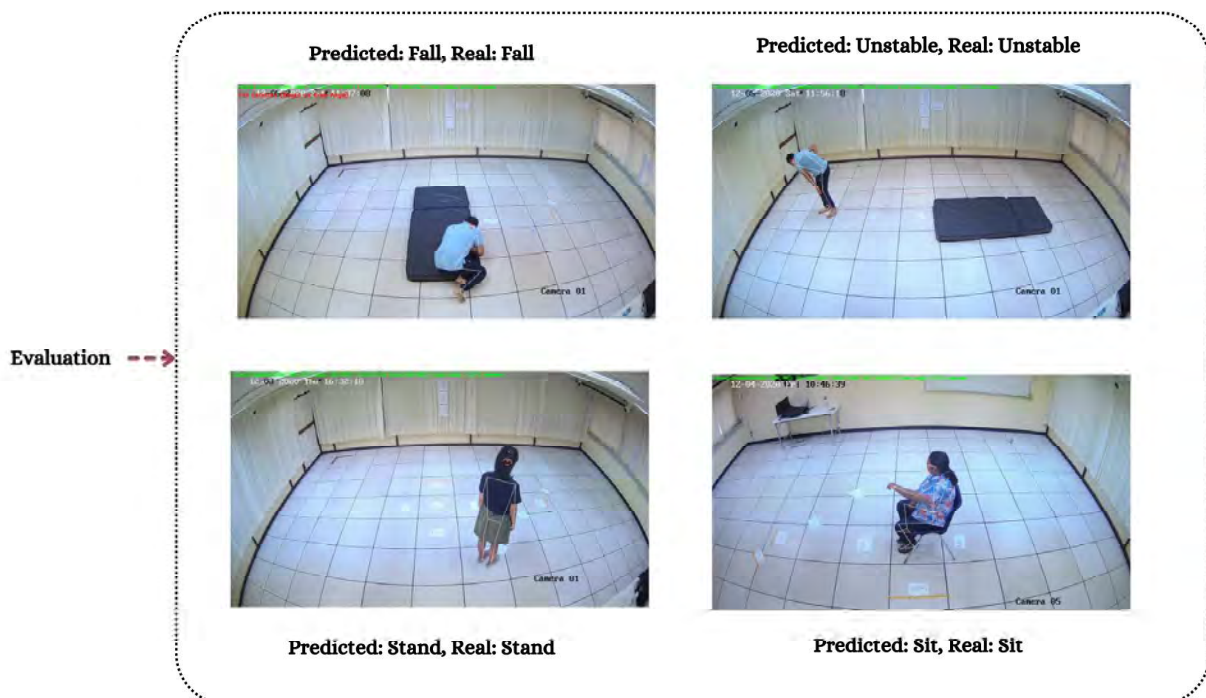
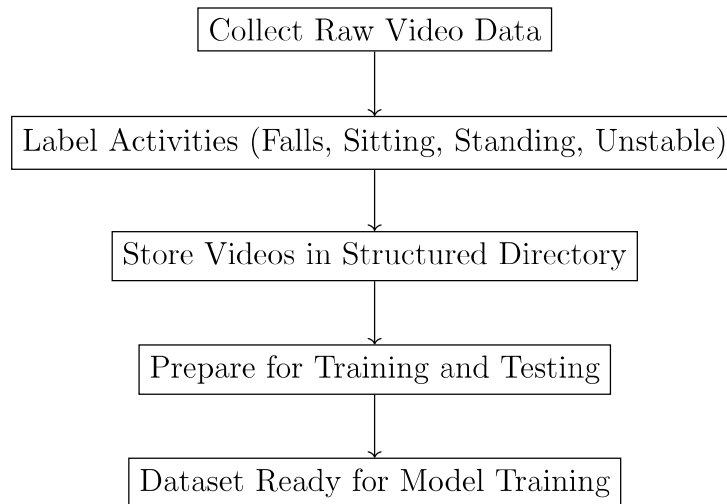


Figure 3.3: Evaluation results showing predicted and actual labels for the four activities.

3.1 Dataset Preparation

For our research, we utilized a dataset from the paper [33]. This dataset comprises four activity classes: Sit, Stand, Unstable, and Fall, and was recorded using accurate CCTV setups in actual homes. The dataset offers valuable insights into daily and recreational activities, enabling a robust classification of behaviours. The dataset is a vital methodology component. It comprises video sequences representing different activities: falls, sitting, standing, and unstable movements. Each activity is associated with specific video files stored in a structured directory, ensuring ease of access and organization for the training and testing processes.



3.1.1 Dataset Preparation for Traditional Machine Learning Classifiers

Data Compilation: The system begins by reading the video paths and corresponding labels from the structured directory. Each video is processed frame by frame to extract the previously defined features. This step builds a comprehensive dataset that captures a wide range of human activities, crucial for training robust classifiers.

Scaling and Splitting: To ensure that all features contribute equally to the model's performance, the extracted features undergo standardization using the StandardScaler method. This normalization step mitigates the influence of differing scales across features, allowing for more effective learning by the classifiers. The dataset is then split into training and testing sets using a 70/30 ratio, a common practice to evaluate the classifier's generalization capabilities. To maintain the distribution of class labels, a stratified method is employed for the train-test split. This ensures that each class is represented proportionally in both the training and testing sets. The training set is used to train the models, while the testing set measures how well the models perform on unseen data.

3.1.2 Dataset Preparation for CNN-LSTM

The dataset preparation for the CNN-LSTM model differs from traditional machine learning classifiers. The CNN-LSTM architecture requires sequences of frames to capture temporal dynamics in activities. Below is an overview of the procedure followed for this model.

1. **Video Processing:** Each video input function captures each video, where frames are read sequentially. The number of frames required for each sequence is predefined; in this case, it is set to 20.
2. **Feature Extraction:** As frames are read, specific features, including pose landmarks, are extracted, which is crucial for understanding human activities. The video frames are resized to a standard dimension (640×480), and landmarks are drawn for visual analysis.
3. **Frame Normalization:** Each frame is resized to 64×64 pixels.
4. **Sequence Creation:** A queue of frames is maintained, allowing the model to create sequences of 20 frames for analysis. This temporal context is critical for the CNN-LSTM model to learn patterns over time. The sequences may overlap since the most recent frames are continuously added to the queue.

In the dataset, several cameras are provided at each location. Later, with each camera's video cropped into short clips corresponding to different activities. The clips were processed frame by frame to extract human poses and create skeleton sequences labelled and stored for training and testing. Figure 3.4 shows one snapshot from each activity class.

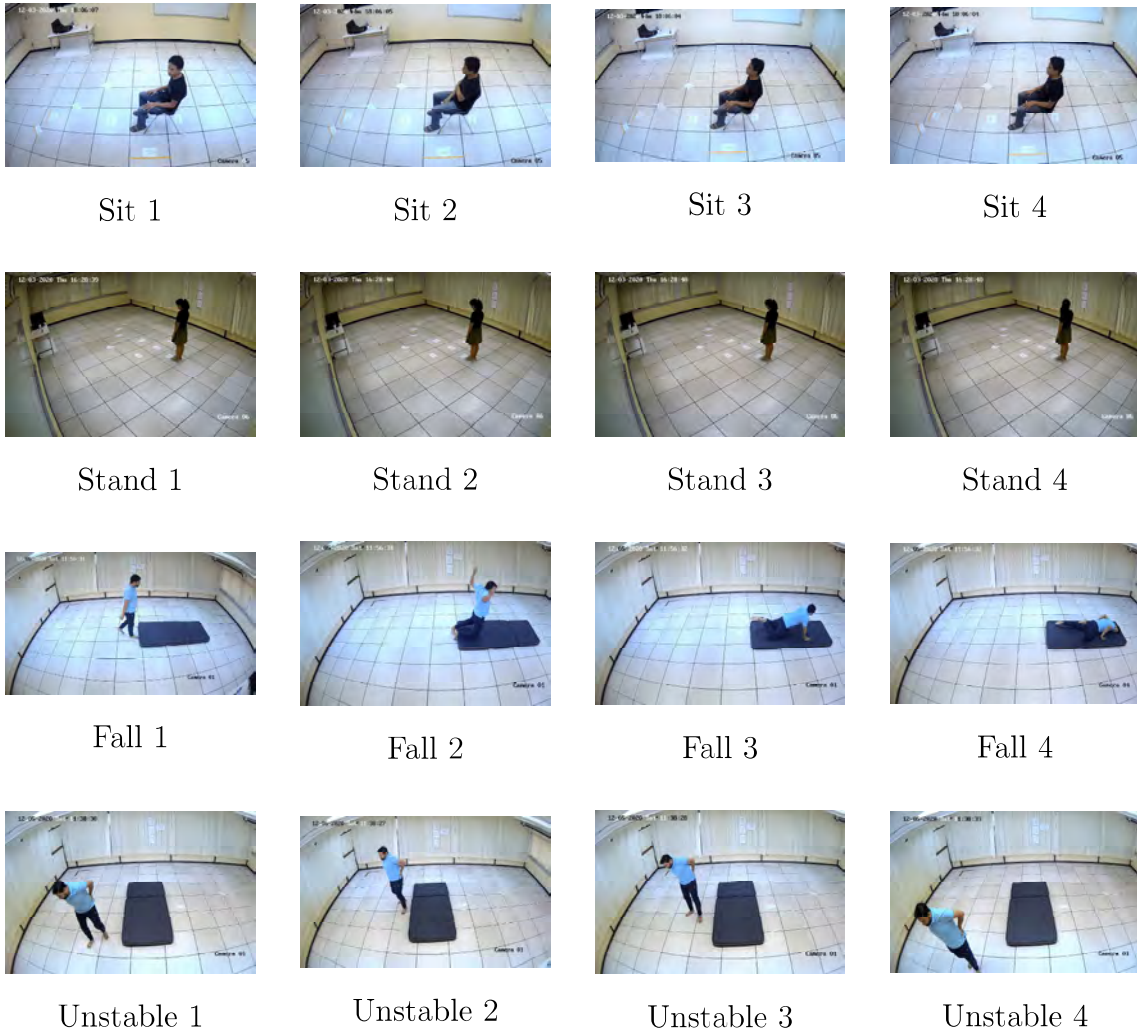


Figure 3.4: Snapshots of Activity Classes from the Dataset

3.2 Feature Extraction

Feature extraction is a critical component of the methodology, which involves deriving meaningful features from the video frames to facilitate effective classification. The system employs the MediaPipe library, specifically its Pose module, to identify and track key human landmarks (e.g., shoulders, hips, knees, and ankles) within the frames. This landmark detection is essential for understanding the subject's posture and movements.

3.2.1 Landmark Detection

The MediaPipe Pose model processes each frame to extract a set of 33 landmarks that provide a comprehensive representation of the human body. These landmarks include:

The model output contains normalized coordinates (Landmarks) and world coordinates (WorldLandmarks) for each landmark, enabling detailed analysis of body orientation and movement dynamics. This detection utilizes a lightweight neural network model designed for high-performance real-time applications, making it particularly effective for tasks that require quick and accurate processing. The coor-

Table 3.1: MediaPipe Pose Landmarks

0 - Nose	1 - Left eye (inner)	2 - Left eye	3 - Left eye (outer)
4 - Right eye (inner)	5 - Right eye	6 - Right eye (outer)	7 - Left ear
8 - Right ear	9 - Mouth (left)	10 - Mouth (right)	11 - Left shoulder
12 - Right shoulder	13 - Left elbow	14 - Right elbow	15 - Left wrist
16 - Right wrist	17 - Left pinky	18 - Right pinky	19 - Left index
20 - Right index	21 - Left thumb	22 - Right thumb	23 - Left hip
24 - Right hip	25 - Left knee	26 - Right knee	27 - Left ankle
28 - Right ankle	29 - Left heel	30 - Right heel	31 - Left foot index
32 - Right foot index			

Coordinates of these landmarks serve as a basis for feature calculation and allow for a more nuanced understanding of the activities being performed. Figure 3.5 illustrates the extracted landmarks. Figure 3.6 illustrates the snapshots of various activities, including falls, sitting, standing, and unstable movements, along with landmarks that aid in the recognition process.

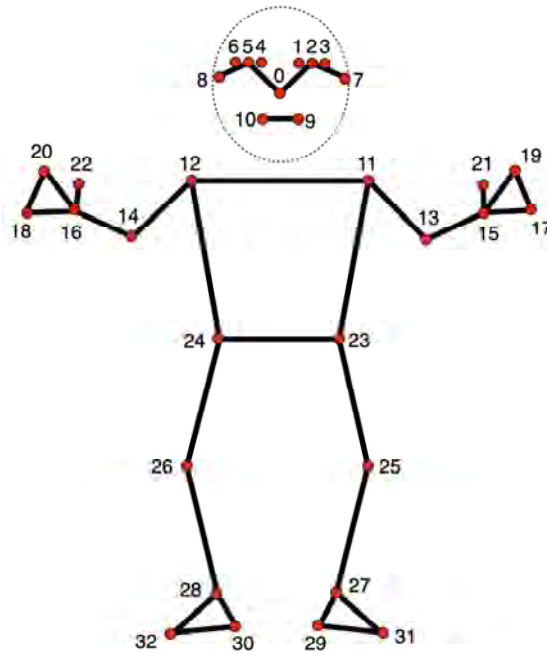


Figure 3.5: Visualization of MediaPipe Pose Landmarks

Falls



Fall 1

Fall 2

Fall 3

Fall 4

Sitting



Sit 1

Sit 2

Sit 3

Sit 4

Standing



Stand 1

Stand 2

Stand 3

Stand 4

Unstable



Unstable 1

Unstable 2

Unstable 3

Unstable 4

Figure 3.6: Dataset Preparation: Snapshots of Activities after applying Mediapipe

3.3 Traditional Machine Learning Classifiers

A series of traditional classifiers are employed to create a baseline for comparison against deep learning techniques. Each classifier is meticulously selected for its proven effectiveness in classification tasks and is subject to rigorous tuning to optimize performance. The classifiers utilized in this study are as follows:

3.3.1 Random Forest Classifier

The Random Forest classifier utilizes an ensemble approach, aggregating predictions from multiple decision trees to enhance classification accuracy and mitigate overfitting. Hyperparameter tuning is conducted using grid search techniques, focusing on optimizing key parameters such as the number of estimators (the count of trees in the forest) and the maximum depth of each tree. This systematic exploration of hyperparameter combinations enables the model to find an optimal balance between complexity and predictive power.

Trainable Parameters: The Random Forest model does not have traditional trainable parameters like weights. Instead, the complexity can be influenced by the number of trees and their maximum depth.

3.3.2 Support Vector Machine (SVM)

SVMs are particularly effective in high-dimensional spaces. To determine the optimal configuration, a grid search is executed to identify the best values for the regularization parameter (which governs the trade-off between maximizing the margin and minimizing classification error) and the kernel type (linear, polynomial, or radial basis function). This tuning process is critical for the SVM to separate the various activity classes effectively.

Trainable Parameters: SVM models have a support vector count that depends on the dataset and parameters selected rather than fixed trainable parameters.

3.3.3 K-Nearest Neighbors (KNN)

The KNN classifier is a non-parametric method that classifies samples based on the labels of their nearest neighbours in the feature space. Optimal performance is achieved through grid search by tuning parameters such as the number of neighbours and the weight function (whether to assign equal weight to all neighbours or prioritize closer ones). KNN's simplicity and interpretability make it valuable to the classifier ensemble.

Trainable Parameters: KNN does not have trainable parameters in the traditional sense; however, the primary hyperparameter is the number of neighbours (k).

3.4 Voting Classifier

A voting classifier has been implemented to enhance classification accuracy further. This ensemble method aggregates predictions from the individual classifiers

discussed above. By utilizing majority voting or averaging probabilities, the voting classifier aims to leverage the strengths of each classifier, improving the overall robustness and reliability of the predictions.

3.5 Deep Learning with CNN-LSTM

While traditional machine learning classifiers establish a solid foundation, deep learning approaches—specifically those employing CNN-LSTM architectures—provide advanced capabilities for capturing intricate patterns in temporal data. The following outlines the methodology for implementing a CNN-LSTM model designed explicitly for activity classification.

3.5.1 Feature Extraction for CNN

In the initial phase of the deep learning pipeline, each frame within a video sequence is resized to standard dimensions and normalized to ensure consistent input data. Normalization enhances convergence during training by scaling pixel values into a manageable range. The preprocessed frames are then input into the CNN model, which excels at extracting spatial features, such as edges and textures.

3.5.2 Model Architecture

The architecture of the CNN-LSTM model is designed to optimize its ability to comprehend spatial and temporal features. The model initiates with several convolutional layers that extract spatial features from the input frames. These layers apply filters to the input data, generating feature maps that emphasize the most informative aspects of the frames. Following the convolutional layers, LSTM layers are integrated to capture the temporal dynamics of the video sequences. LSTMs are particularly advantageous in this context due to their capability to retain information over extended sequences, effectively addressing the vanishing gradient problem often faced by traditional RNNs.

The architecture culminates in fully connected layers that synthesize the learned features and output the predicted activity class. The final output layer employs a softmax activation function to yield probabilities for each activity class, facilitating definitive classification. Figure 3.7 shows the architecture of custom built of CNN-LSTM.

Trainable Parameters: The CNN-LSTM model has a total of 7894468 trainable parameters, depending on the number of layers, filters, and units defined in the architecture.

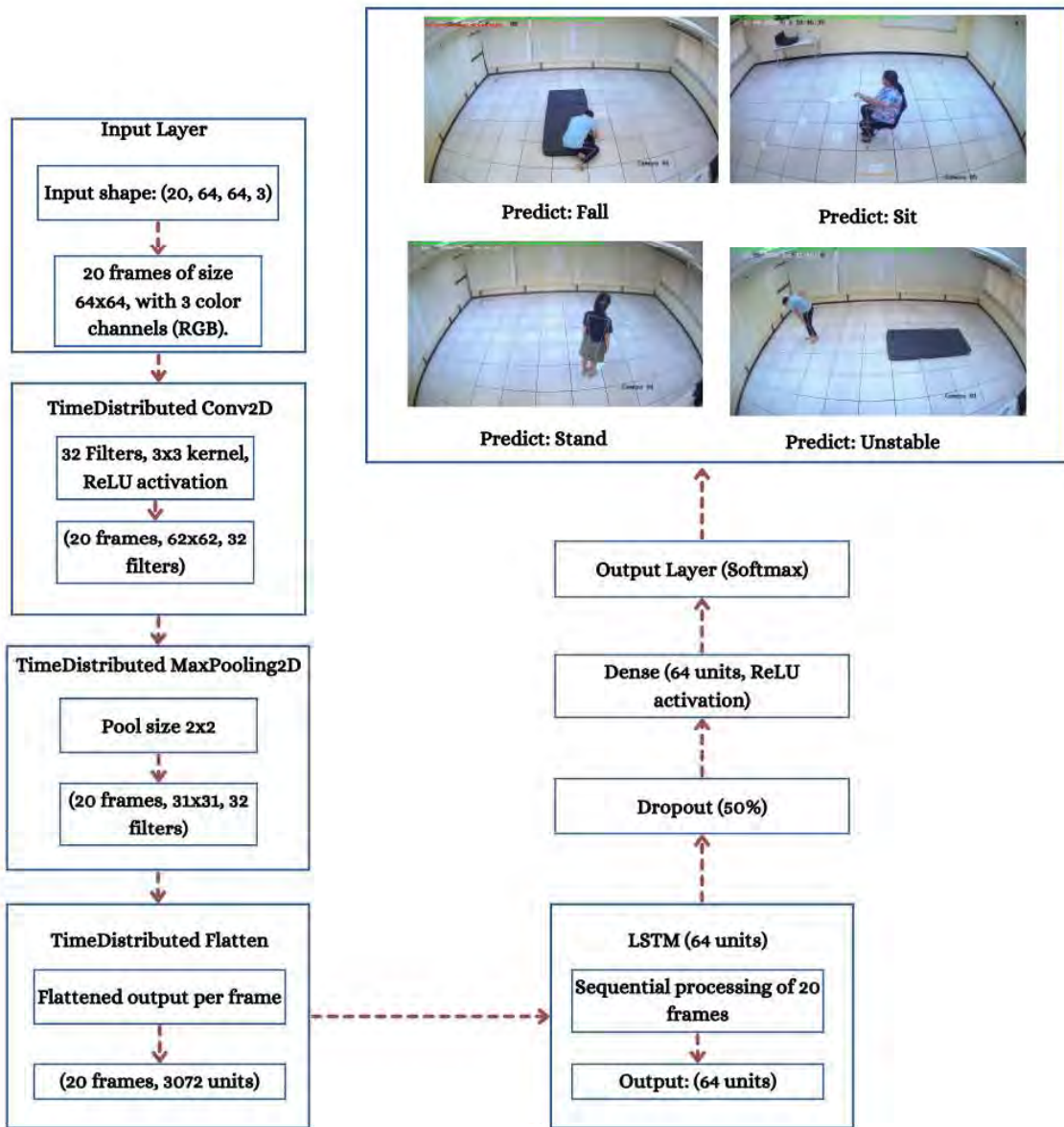


Figure 3.7: CNN-LSTM architecture for human activity recognition.

3.6 Enhanced Features

The foundation of our system lies in video processing, which initiates with capturing frames from the input video. Leveraging the OpenCV library—a robust tool in computer vision—our system efficiently reads and processes each frame in real-time. This capability is crucial for applications that demand immediate feedback, particularly in safety monitoring systems where prompt responses are essential. In this context, the system employs advanced techniques for activity recognition despite additional training, such as detecting falls through knee angle assessment and analyzing body posture via angle and distance calculations. Integrating these methodologies ensures a comprehensive approach to monitoring and enhancing safety in dynamic environments.

3.6.1 Fall Detection Using Knee Angle

My system also detects falls by utilizing knee angles as a critical metric. In cases where the upper body is not visible to the camera, we rely on the angle formed at the knee joint to identify potential falls. The knee angle is calculated using the hip, knee, and ankle landmarks.

The system is in action in the Figure 3.8. The person is on the ground, and the fall detection is based on the knee angle, which is indicated as 44.75 degrees. The key points on the person’s body—especially the lower body—are tracked, and the system has triggered a 'fall detected' warning due to the knee angle.

To calculate the knee angle, the system uses the following equation, derived from the law of cosines:

$$\theta_{\text{knee}} = \cos^{-1} \left(\frac{(d_{\text{hip-knee}})^2 + (d_{\text{knee-ankle}})^2 - (d_{\text{hip-ankle}})^2}{2 \cdot d_{\text{hip-knee}} \cdot d_{\text{knee-ankle}}} \right)$$

Where:

- $d_{\text{hip-knee}}$ is the distance between the hip and knee landmarks,
- $d_{\text{knee-ankle}}$ is the distance between the knee and ankle landmarks,
- $d_{\text{hip-ankle}}$ is the distance between the hip and ankle landmarks.

If the knee angle θ_{knee} is less than 70 degrees, a fall is detected. Based on the visual data and the algorithm’s processing, the knee angle was measured at 44.75 degrees, which led to fall detection.



Figure 3.8: Fall based on angle.

3.6.2 Angle and Distance Calculation

Following landmark detection, the system computes various angles and distances to quantify the subject's posture and movement. These features are derived using Euclidean distance and trigonometric formulas.

The Euclidean distance between two landmarks $A(x_1, y_1, z_1)$ and $B(x_2, y_2, z_2)$ is given by Equation (3.1):

$$d_{AB} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \quad (3.1)$$

For instance, the shoulder-hip distance, which calculates the distance between the left shoulder and left hip landmarks, is computed using Equation (3.2):

$$d_{\text{shoulder-hip}} = \sqrt{(x_{\text{hip}} - x_{\text{shoulder}})^2 + (y_{\text{hip}} - y_{\text{shoulder}})^2 + (z_{\text{hip}} - z_{\text{shoulder}})^2} \quad (3.2)$$

Similarly, the shoulder-nose distance is calculated as shown in Equation (3.3):

$$d_{\text{shoulder-nose}} = \sqrt{(x_{\text{nose}} - x_{\text{shoulder}})^2 + (y_{\text{nose}} - y_{\text{shoulder}})^2 + (z_{\text{nose}} - z_{\text{shoulder}})^2} \quad (3.3)$$

To compute angles between three landmarks $A(x_1, y_1)$, $B(x_2, y_2)$, and $C(x_3, y_3)$, the angle θ at point B is derived using the law of cosines (Equation (3.4)):

$$\theta = \cos^{-1} \left(\frac{(d_{AB})^2 + (d_{BC})^2 - (d_{AC})^2}{2 \cdot d_{AB} \cdot d_{BC}} \right) \quad (3.4)$$

For instance, the shoulder angle, defined as the angle at the shoulder between the left elbow, left shoulder, and left hip landmarks, is computed using Equation (3.5):

$$\theta_{\text{shoulder}} = \cos^{-1} \left(\frac{(d_{\text{elbow-shoulder}})^2 + (d_{\text{shoulder-hip}})^2 - (d_{\text{elbow-hip}})^2}{2 \cdot d_{\text{elbow-shoulder}} \cdot d_{\text{shoulder-hip}}} \right) \quad (3.5)$$

To normalize features and ensure robustness, the system also computes ratios of distances. For example, the ratio of the shoulder-hip distance to the shoulder-nose distance is given by Equation (3.6):

$$\text{Ratio}_{\text{shoulder-hip:nose}} = \frac{d_{\text{shoulder-hip}}}{d_{\text{shoulder-nose}}} \quad (3.6)$$

3.7 Implementation Aspects

The complete methodology has been implemented in Python, leveraging a range of libraries and tools to facilitate the various stages of the activity classification process. This chapter provides an in-depth overview of the implementation aspects, including the libraries used, the hyperparameter tuning process, and the evaluation metrics employed to assess model performance.

3.7.1 Libraries and Tools

The implementation utilizes several key libraries that are integral to the functionality of the system:

- **OpenCV:** This powerful computer vision library is employed for video processing tasks, including capturing video streams, reading frames, and applying various image transformations. OpenCV's efficient handling of video data enables real-time processing, which is crucial for the timely classification of human activities.
- **MediaPipe:** For pose detection, the MediaPipe library is utilized, particularly its Pose module. This library offers state-of-the-art capabilities in real-time human pose estimation by detecting critical landmarks on the human body. The accuracy and speed of MediaPipe make it an ideal choice for feature extraction in this project.
- **Scikit-learn:** This library implements traditional machine learning algorithms. Scikit-learn provides a user-friendly interface for various classifiers, including Random Forest, Support Vector Machine, and K-Nearest Neighbors. Its functionalities include model training, hyperparameter tuning through grid search, and performance evaluation.
- **TensorFlow/Keras:** For the deep learning components, TensorFlow and Keras are employed to build and train the CNN-LSTM model. As an API for TensorFlow, Keras simplifies the creation of neural network architectures and facilitates easy experimentation with different configurations.

3.7.2 Hyperparameter Tuning

Hyperparameter tuning is a critical step in optimizing the performance of machine learning models. The process involves systematically searching for the best hyperparameter configurations that yield the highest accuracy on the validation dataset. The following steps outline the hyperparameter tuning process employed in this project:

- **Grid Search:** For each classifier, a grid search is conducted over a predefined set of hyperparameters. For instance, the Random Forest Classifier explores parameters such as the number of trees, maximum depth, and minimum samples per leaf. Similarly, different kernel types and regularization parameters are evaluated for the SVM.
- **Cross-Validation:** To ensure robustness in the evaluation of hyperparameter combinations, k-fold cross-validation is employed. This technique splits the dataset into k subsets, training the model on k-1 subsets while validating the remaining subset. The process is repeated k times, and the average performance is calculated to provide a reliable estimate of model effectiveness.
- **Best Model Selection:** After completing the grid search, the best model configuration is selected based on the highest validation accuracy. This ensures that the model is fine-tuned for optimal performance on unseen data.

3.7.3 Model Training and Parameters

The model training process included applying three machine learning classifiers: RF, SVM, and KNN. For each classifier, hyperparameter tuning was conducted using Grid Search with threefold cross-validation.

- **Random Forest:** The optimal hyperparameters were found to be `n_estimators=100`, `max_depth=None`, and `min_samples_split=2`.
- **Support Vector Machine:** The best combination was `C=1`, `kernel=rbf`, and `gamma=scale`.
- **K-Nearest Neighbors:** The optimal hyperparameters were `n_neighbors=5` and `weights=uniform`.

After training these individual classifiers, a soft-voting ensemble classifier was constructed. Due to its superior individual performance, the Random Forest model was weighted twice as heavily as the SVM and KNN models. The ensemble model was trained on the dataset using 80% of the data for training and 20% for testing.

3.7.4 Evaluation Metrics

Once the models are trained and optimized, their performance is evaluated using a variety of metrics to assess their effectiveness in classifying human activities. The evaluation process includes the following components:

- **Accuracy:** The overall accuracy of each classifier is calculated as the ratio of correctly predicted instances to the total instances in the test dataset. This metric provides a straightforward assessment of model performance.
- **Confusion Matrix:** The confusion matrix is utilized to visualize the performance of the classification models. It displays the true positive, true negative, false positive, and false negative counts, allowing for a detailed examination of how well each class is recognized. This matrix aids in identifying specific classes that may be misclassified more frequently.
- **Classification Report:** A comprehensive classification report is generated for each model, which includes precision, recall, and F1-score metrics for each activity class. Precision indicates the proportion of accurate positive predictions among all optimistic predictions, while recall measures the ability of the model to identify all relevant instances. The F1 score provides a balance between precision and recall, which is particularly useful for imbalanced datasets.
- **ROC Curves:** ROC curves are plotted to illustrate the trade-off between sensitivity (actual positive rate) and specificity (false positive rate) at various threshold settings. The area under the ROC curve (AUC) is calculated to quantify the model's ability to distinguish between classes, with higher values indicating better performance.

This comprehensive evaluation process ensures that the models learn the underlying patterns within the data and generalize effectively to novel, unseen activity sequences.

Chapter 4

Results and Performance Analysis

This chapter presents the results and performance analysis of various models implemented for human activity recognition. The models were evaluated based on several performance metrics: accuracy, confusion matrix, ROC curves, precision, recall, and F1 scores. These metrics comprehensively understand how well each model performed across the different activity classes. Comparisons between traditional machine learning models like Random Forest, Support Vector Machine, K-Nearest Neighbors, Voting Classifier, and the deep learning CNN-LSTM model are also provided.

4.0.1 Comparative Analysis of Model Performance

The performance of each model was assessed using accuracy scores, confusion matrices, and classification reports. Table 4.1 summarizes the accuracy achieved by different models.

Table 4.1: Accuracy of Different Models

Model	Accuracy (%)
Random Forest	97.0
Support Vector Machine	95.0
K-Nearest Neighbors	98.0
Voting Classifier	98.0
CNN-LSTM	98.0

The Voting Classifier(K-Nearest Neighbors, Support Vector Machine, Random Forest combined) has an accuracy of 98.0%. The CNN-LSTM model also achieved 98.0%, showcasing its ability to capture both spatial and temporal dependencies in video data.

4.0.2 Confusion Matrix

The normalized confusion matrix for Voting Classifier is shown in Figure 4.1. The model was highly accurate in classifying the activities, with the most notable misclassification occurring between the Unstable and Fall classes.

Lastly, the confusion matrix for the CNN-LSTM model is presented in Figure 4.2. This model demonstrated higher precision across all classes.

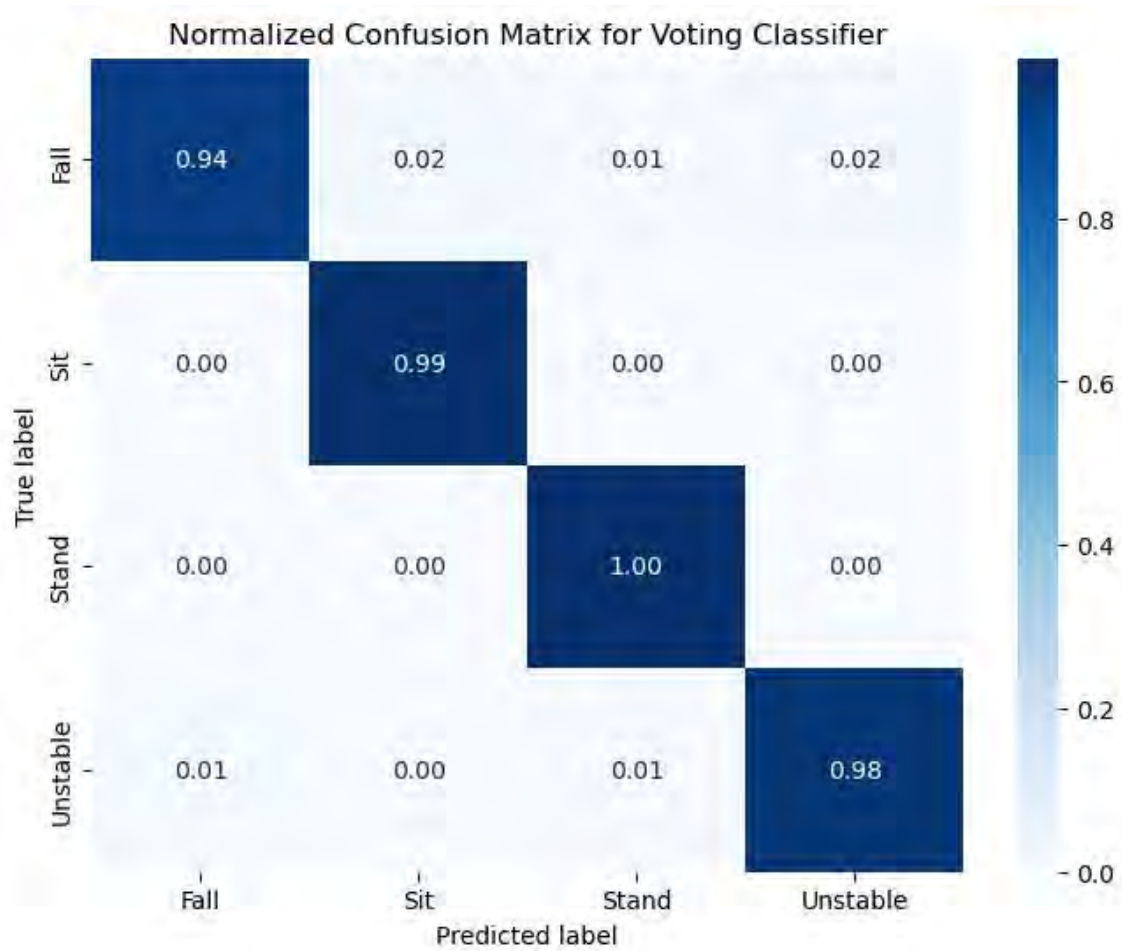


Figure 4.1: Normalized Confusion Matrix for Voting Classifier.

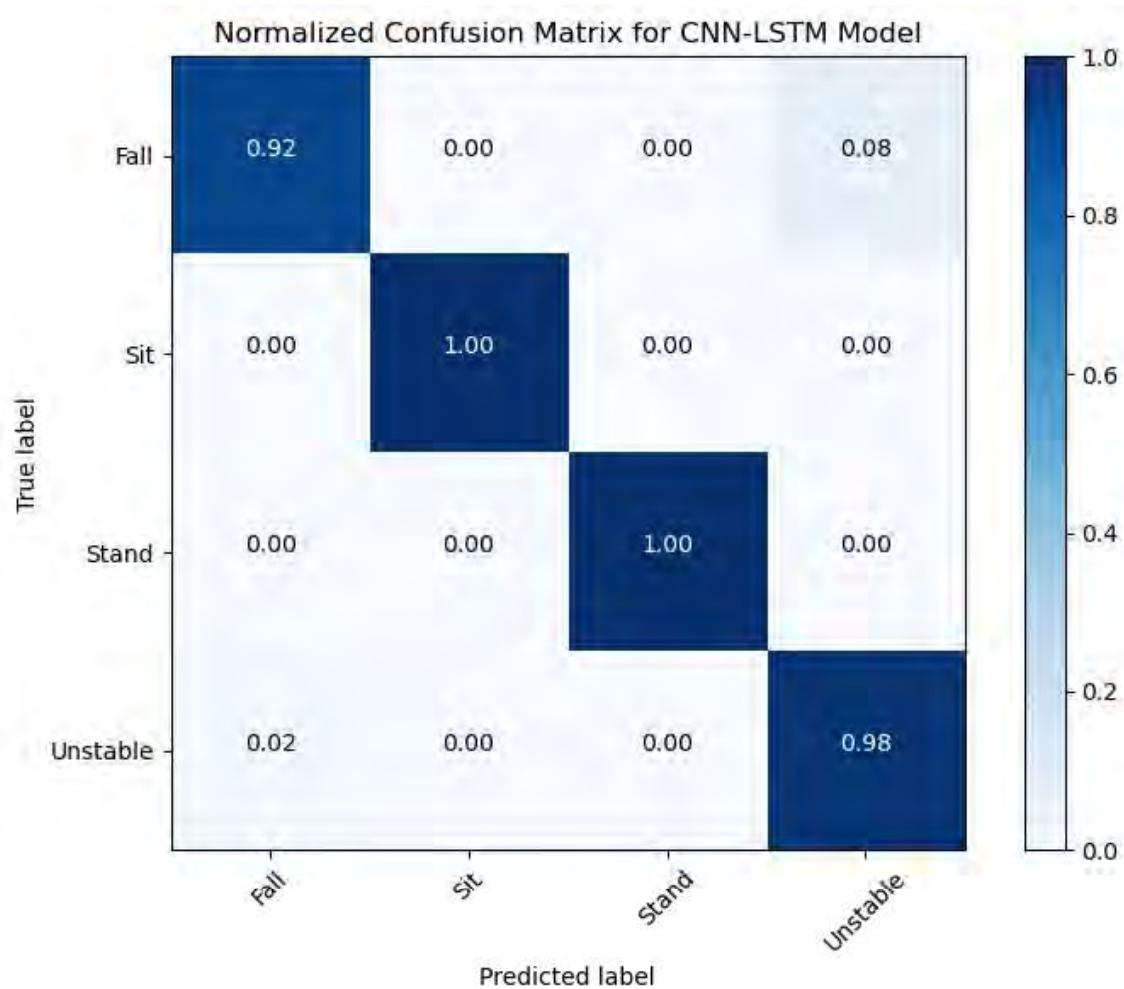


Figure 4.2: Normalized Confusion Matrix for CNN-LSTM Model.

4.0.3 ROC Curves

To further evaluate the models, ROC curves were generated. The area under the curve AUC serves as a key performance metric. The ROC curves for the Voting Classifier are displayed in Figure 4.3, while the CNN-LSTM model's ROC curves are shown in Figure 4.4.

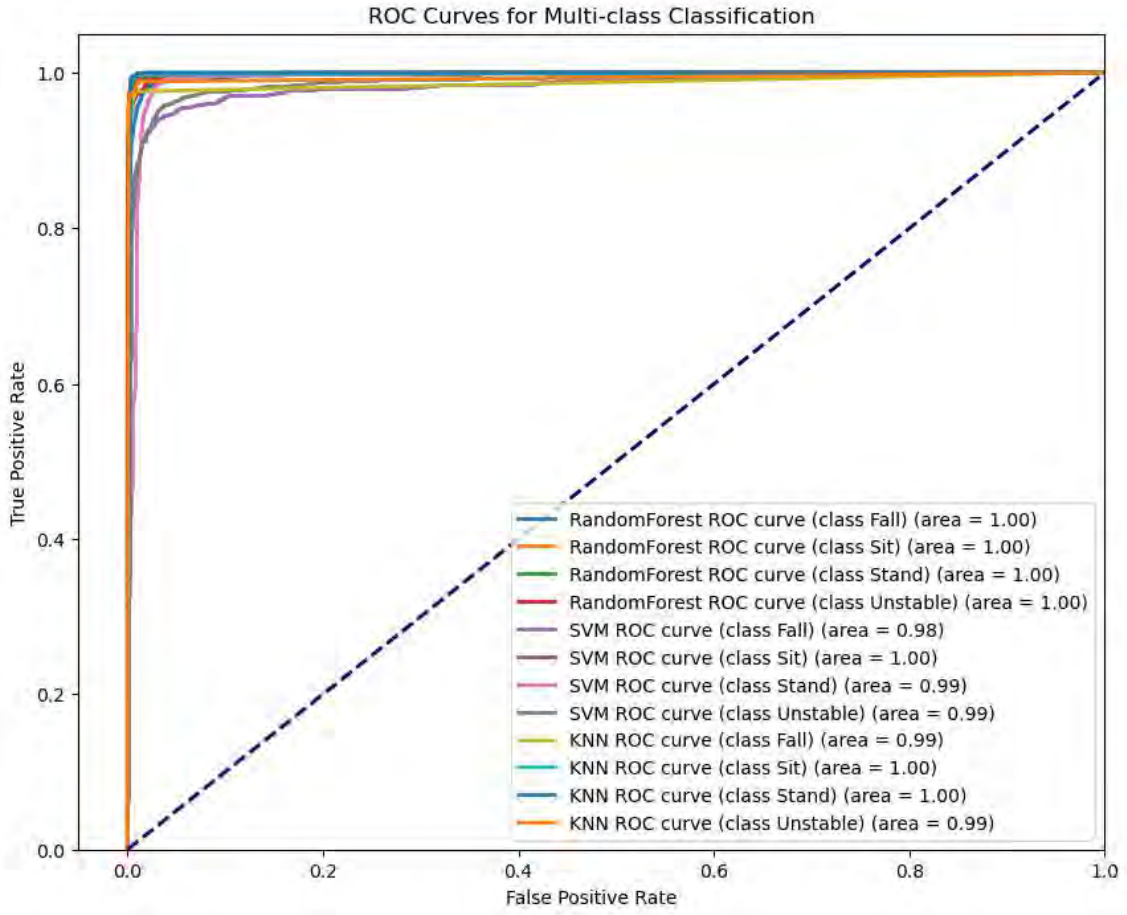


Figure 4.3: ROC Curves for Voting Classifier.

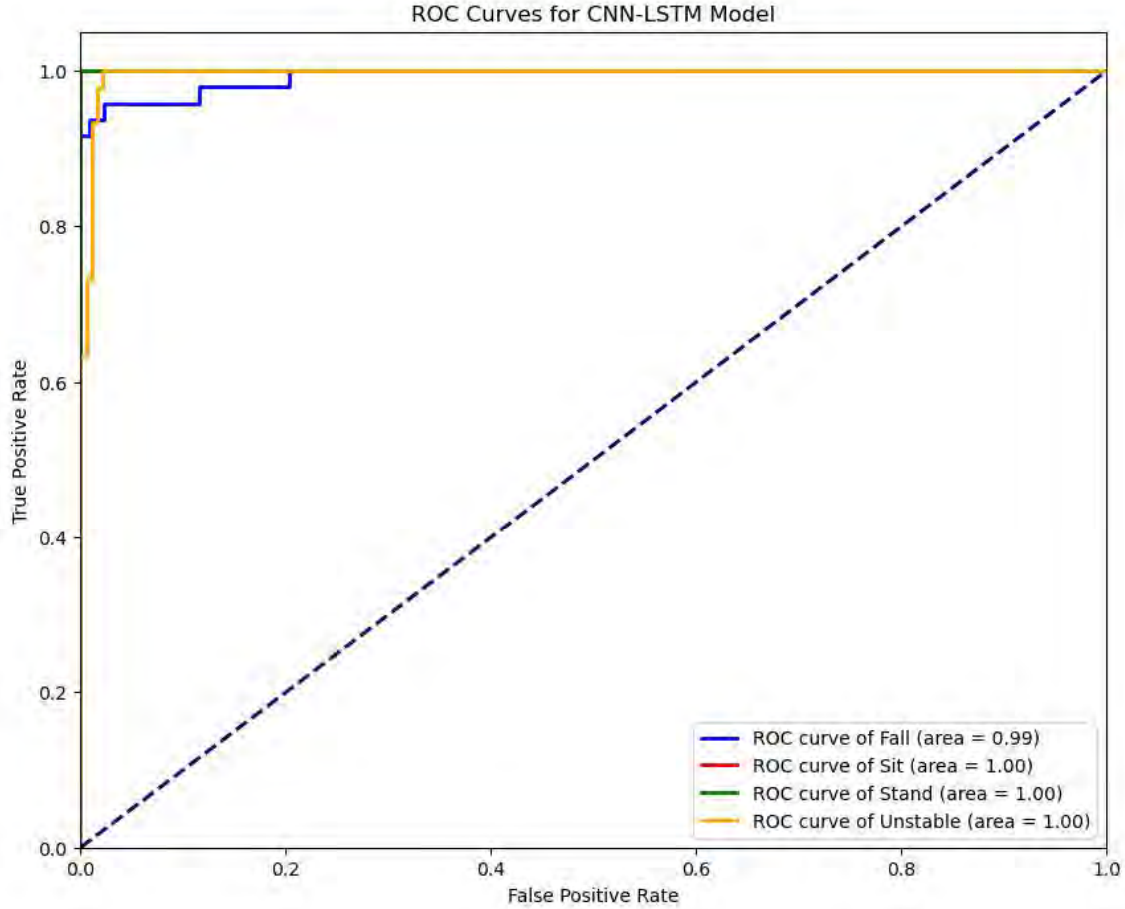


Figure 4.4: ROC Curves for CNN-LSTM Model.

4.0.4 Classification Results

Tables 4.2 and 4.3 provide the classification reports for the Voting Classifier and the CNN-LSTM model, respectively. These reports include the precision, recall, and F1-score for each class.

Table 4.2: Classification Report for Voting Classifier

Class	Precision	Recall	F1-Score
Fall	0.98	0.94	0.96
Sit	0.98	0.99	0.99
Stand	0.97	1.0	0.98
Unstable	0.98	0.98	0.98

Table 4.3: Classification Report for CNN-LSTM Model

Class	Precision	Recall	F1-Score
Fall	0.96	0.92	0.94
Sit	1.00	1.00	1.00
Stand	1.00	1.00	1.00
Unstable	0.96	0.98	0.97

The CNN-LSTM model achieved high precision especially in complex activities like 'Unstable' and 'Fall', reflecting its robustness in recognizing time-sensitive actions.

4.0.5 Visual Representation of Classified Activities

This section showcases visual representations of the outcomes generated by the Voting Classifier and the CNN-LSTM model for human activity recognition, which includes knee angle measurements. The images depict the various classified activities: (a) Fall (Figure 4.5), (b) Unstable (Figure 4.6), (c) Stand (Figure 4.7), and (d) Sit (Figure 4.8). These visuals emphasize the model's effectiveness in accurately detecting and classifying the four essential activities within the video sequences.



Figure 4.5: A fall incident forecasted by the trained model.



Figure 4.6: An unstable condition predicted by the trained model.



Figure 4.7: A standing posture predicted by the trained model.



Figure 4.8: A sitting posture forecasted by the trained model.

Chapter 5

Conclusion

In this paper, we explored the performance of traditional machine learning models such as Random Forest, Support Vector Machine, and K-Nearest Neighbors, as well as a CNN-LSTM model for human activity recognition, particularly for detecting falls and unstable postures. The CNN-LSTM model successfully captured the temporal dependencies inherent in video sequences, making it particularly well-suited for activity recognition tasks.

While the results of this study are promising, certain limitations need to be acknowledged. The current implementation may struggle in low-light scenarios, decreasing fall detection and activity recognition accuracy. This limitation emphasizes the need for advanced video preprocessing techniques or infrared cameras to maintain performance in dark environments. Additionally, the computational requirements for the CNN-LSTM model may limit its real-time processing capabilities on less powerful devices, potentially affecting the system's responsiveness in resource-constrained environments. Furthermore, the model was trained and tested on two to three persons, which may result in performance degradation when more than four persons are in the video footage. This necessitates further validation and training on diverse datasets to improve robustness in crowded or dynamic settings.

There are several areas where future work is planned to enhance the system. One of the key objectives is to implement the entire human activity recognition system on an NVIDIA Jetson device. The Jetson platform's ability to perform real-time video processing and its support for deep learning models make it suitable for edge-based fall detection and activity recognition. Additionally, we plan to develop a web-based API that can be accessed via a website built using Next.js and a mobile application using Flutter. This will allow caregivers or medical professionals to monitor individuals' activities in real time, providing notifications, activity logs, and emergency alerts. The system will also be equipped with an alert mechanism that sends automatic notifications via SMS and the mobile app if the target person falls or remains in an unstable posture for an extended period. This notification system will be critical in ensuring timely assistance and reducing the risks associated with falls and prolonged unstable states. Lastly, to ensure the system works effectively in low-light conditions, we plan to add a night mode feature that will use advanced video preprocessing techniques to detect falls in the dark. This enhancement may involve infrared cameras or other low-light vision techniques to ensure that fall detection remains accurate during nighttime or low-light scenarios.

In conclusion, our future work will focus on improving the system's real-time respon-

siveness and accessibility while ensuring robustness across various environments, including low-light conditions.

Bibliography

- [1] Tarek Al-Aama. “Falls in the elderly”. In: *Canadian Family Physician* 57.7 (2011), pp. 771–776. URL: <https://www.cfp.ca/content/57/7/771>.
- [2] T. Lin, C. Yang, and W. Shih. “Fall prevention shoes using camera-based line-laser obstacle detection system”. In: *Journal of Healthcare Engineering* 2017 (2017), pp. 1–11. DOI: 10.1155/2017/8264071. URL: <https://doi.org/10.1155/2017/8264071>.
- [3] M. Mubashir, L. Shao, and L. A. Seed. “Survey on fall detection: principles and approaches”. In: *Neurocomputing* 100 (2013), pp. 144–152. DOI: 10.1016/j.neucom.2011.09.037. URL: <https://doi.org/10.1016/j.neucom.2011.09.037>.
- [4] K. Ali, A. Mian, and M. Hayat. “A survey of fall detection approaches”. In: *Artificial Intelligence Review* 54.7 (2021), pp. 4577–4617. DOI: 10.1007/s10462-021-09926-3. URL: <https://doi.org/10.1007/s10462-021-09926-3>.
- [5] D. Anderson et al. “Recognizing falls from silhouettes”. In: *Proceedings of the 2006 International Conference of the IEEE Engineering in Medicine and Biology Society*. 2006, pp. 6388–6391. DOI: 10.1109/IEMBS.2006.259594. URL: <https://doi.org/10.1109/IEMBS.2006.259594>.
- [6] Z. Zhou, W. Dai, J. Eggert, et al. “A real-time system for in-home activity monitoring of elders”. In: *Proceedings of the 2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. 2009, pp. 6115–6118. DOI: 10.1109/IEMBS.2009.5334915. URL: <https://doi.org/10.1109/IEMBS.2009.5334915>.
- [7] P. C. Chung and C. D. Liu. “A daily behavior enabled hidden Markov model for human behavior understanding”. In: *Pattern Recognition* 41.5 (2008), pp. 1572–1580. DOI: 10.1016/j.patcog.2007.10.022. URL: <https://doi.org/10.1016/j.patcog.2007.10.022>.
- [8] Z. Zhou et al. “Activity analysis, summarization, and visualization for indoor human activity monitoring”. In: *IEEE Transactions on Circuits and Systems for Video Technology* 18.11 (2008), pp. 1489–1498.
- [9] Y. Wang and H. Wu. “Real-time fall detection system using a smartphone”. In: *Proceedings of the 2015 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. 2015, pp. 148–153. DOI: 10.1109/PERCOMW.2015.7137715. URL: <https://doi.org/10.1109/PERCOMW.2015.7137715>.
- [10] T. Zhou et al. “MATNet: Motion-attentive transition network for zero-shot video object segmentation”. In: *IEEE Transactions on Image Processing* (2020), pp. 8326–8338.

- [11] E. E. Stone and M. Skubic. “Evaluation of an inexpensive depth camera for passive in-home fall risk assessment”. In: *Proceedings of the 2011 5th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth) and Workshops*. 2011, pp. 71–77. DOI: 10.4108/icst.pervasivehealth.2011.246034. URL: <https://doi.org/10.4108/icst.pervasivehealth.2011.246034>.
- [12] D. Anguita et al. “A public domain dataset for human activity recognition using smartphones”. In: *Proceedings of the 21st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*. 2013, pp. 437–442. URL: <http://www.esann.org/proceedings/>.
- [13] A. Shahroudy et al. “NTU RGB+D: A large scale dataset for 3D human activity analysis”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 1010–1019. DOI: 10.1109/CVPR.2016.121. URL: <https://doi.org/10.1109/CVPR.2016.121>.
- [14] H. Wang et al. “Depth appearance-based local occupancy patterns for action recognition”. In: *Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP)*. 2016, pp. 1241–1245. DOI: 10.1109/ICIP.2016.7532545. URL: <https://doi.org/10.1109/ICIP.2016.7532545>.
- [15] J. Wu, Z. Zhang, and W. Li. “A survey on human activity recognition in video”. In: *Computer Vision and Image Understanding* 171 (2018), pp. 1–19. DOI: 10.1016/j.cviu.2018.07.001. URL: <https://doi.org/10.1016/j.cviu.2018.07.001>.
- [16] C. Li et al. “Skeleton-based action recognition with convolutional neural networks”. In: *Proceedings of the 2017 IEEE International Conference on Multimedia Expo Workshops (ICMEW)*. 2017, pp. 597–600. DOI: 10.1109/ICMEW.2017.8026285. URL: <https://doi.org/10.1109/ICMEW.2017.8026285>.
- [17] S. Zhang, X. Liu, and J. Xiao. “On geometric features for skeleton-based action recognition using multilayer LSTM networks”. In: *Proceedings of the 2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2017, pp. 148–157. DOI: 10.1109/WACV.2017.24. URL: <https://doi.org/10.1109/WACV.2017.24>.
- [18] K. Simonyan and A. Zisserman. “Two-stream convolutional networks for action recognition in videos”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 27. 2014. URL: <https://arxiv.org/abs/1406.2199>.
- [19] Y. A. Andrade-Ambriz et al. “Human activity recognition using temporal convolutional neural network architecture”. In: *Expert Systems with Applications* 191 (2022), p. 116287.
- [20] J. Donahue et al. “Long-term recurrent convolutional networks for visual recognition and description”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.4 (2017), pp. 839–854. DOI: 10.1109/TPAMI.2016.2559690. URL: <https://doi.org/10.1109/TPAMI.2016.2559690>.
- [21] M. Hussain, M. I. Qureshi, and I. Hussain. “Real-time human activity recognition using a lightweight deep learning model”. In: *Sensors* 20.15 (2020), p. 4106. DOI: 10.3390/s20154106. URL: <https://doi.org/10.3390/s20154106>.

- [22] Google. “MediaPipe Pose Landmarker”. In: *Google AI Edge Solutions*. 2023. URL: https://ai.google.dev/edge/mediapipe/solutions/vision/pose_landmarker.
- [23] R. Gupta, M. Singh, and A. Prakash. “Human Pose Estimation using MediaPipe and Deep Learning”. In: *Proceedings of the 2021 IEEE 6th International Conference on Image, Vision and Computing (ICIVC)*. 2021, pp. 72–77. DOI: 10.1109/ICIVC52380.2021.00020. URL: <https://doi.org/10.1109/ICIVC52380.2021.00020>.
- [24] S. Kumar, R. Gupta, and P. Kumar. “Real-Time Hand Gesture Recognition using MediaPipe”. In: *Proceedings of the 2022 International Conference on Computational Intelligence and Data Science (ICCIDS)*. 2022, pp. 1–5. DOI: 10.1109/ICCIDS53956.2022.9731640. URL: <https://doi.org/10.1109/ICCIDS53956.2022.9731640>.
- [25] A. Othman, A. Elyazidi, and A. Elhousni. “Pose-Based Action Recognition using MediaPipe and CNN”. In: *Journal of Real-Time Image Processing* 20 (2023), pp. 1389–1401. DOI: 10.1007/s11554-023-01137-0. URL: <https://doi.org/10.1007/s11554-023-01137-0>.
- [26] H. Al-Sadi, M. Al-Adhaileh, and A. Al-Dahoud. “Fitness Monitoring System using MediaPipe for Real-Time Motion Analysis”. In: *2023 International Conference on Computer Applications (ICCA)*. 2023, pp. 1–5. DOI: 10.1109/ICCA57500.2023.10001327. URL: <https://doi.org/10.1109/ICCA57500.2023.10001327>.
- [27] A. Patel, V. Kumar, and R. Desai. “Real-Time Facial Emotion Recognition using MediaPipe”. In: *Proceedings of the 2022 International Conference on Advanced Computer Science and Information Technology (ICACSIT)*. 2022, pp. 34–39. DOI: 10.1109/ICACSIT55965.2022.9731752. URL: <https://doi.org/10.1109/ICACSIT55965.2022.9731752>.
- [28] X. Chen, Y. Lin, and J. Zhang. “Daily Activity Recognition using MediaPipe Skeleton Detection”. In: *2022 International Conference on Image Processing and Computer Vision (IPCV)*. 2022, pp. 88–92. DOI: 10.1109/IPCV56651.2022.00019. URL: <https://doi.org/10.1109/IPCV56651.2022.00019>.
- [29] Y. Liu, T. Wang, and Y. Zhang. “Interactive Virtual Try-On Application using MediaPipe and Augmented Reality”. In: *2023 IEEE International Conference on Virtual Reality and Augmented Reality (VRAR)*. 2023, pp. 45–50. DOI: 10.1109/VRAR54123.2023.10002538. URL: <https://doi.org/10.1109/VRAR54123.2023.10002538>.
- [30] L. Cheng, Y. Wei, and J. Zhao. “Sign Language Recognition using MediaPipe and RNN”. In: *2023 International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, pp. 63–68. DOI: 10.1109/CVPR52688.2023.00018. URL: <https://doi.org/10.1109/CVPR52688.2023.00018>.
- [31] H. Zhang, Y. Li, and X. Zhao. “Smart Home Automation using MediaPipe for Gesture Recognition”. In: *2022 International Conference on Smart Home Technologies (ICSH)*. 2022, pp. 25–30. DOI: 10.1109/ICSH56912.2022.00007. URL: <https://doi.org/10.1109/ICSH56912.2022.00007>.

- [32] Y. Jiang, X. Chen, and J. Li. “Abnormal Activity Detection in Video Surveillance using MediaPipe”. In: *2023 IEEE International Conference on Information and Communication Technology (ICICT)*. 2023, pp. 1–5. DOI: 10.1109/ICICT56023.2023.00009. URL: <https://doi.org/10.1109/ICICT56023.2023.00009>.
- [33] Aniqua Nusrat Zereen et al. “Video analytic system for activity profiling, fall detection, and unstable motion detection”. In: *Proceedings of the 2020 International Conference on Computer Vision and Image Processing*. 2020, pp. 1–6. URL: <https://link.springer.com/article/10.1007/s11042-023-14993-y>.