# Fall Detection from video footage

## 55-708252- AI Research and Development Project


### Supervisor: Dr. Jing Wang

### Student: Mohammed Alsiraji - 33064634


## Introduction

Falls, particularly among the elderly, represent a significant health concern due to the potential for severe injuries and fatalities. The medical consequences are often worsened by prolonged periods of immobility following a fall, making timely detection and intervention critical. The challenge in fall detection lies in accurately identifying falls amidst various daily activities. The difficulty increases by the variability in how falls occur and the need to differentiate them from other activities, such as sitting down or stumbling (Espinosa et al., 2019). Key considerations include detecting the presence of a person, tracking their movement, and accurately interpreting their actions.

There are two primary approaches for fall detection systems: sensor-based and vision-based systems. Sensor-based systems utilise wearable devices, ambient sensors, and smart devices to monitor movements and detect falls based on acceleration and other metrics. However, these systems may suffer from limitations, such as false positives and negatives, leading to unnecessary alarms or missed incidents. In contrast, vision-based systems employ cameras and image processing techniques, often enhanced by deep learning algorithms, to analyse visual data for fall detection.

The final prototype developed utilises a vision-based approach for fall detection. The system employs pose estimation models, specifically pre-trained MediaPipe and MoveNet models, to detect and track body movements in video recordings. These models extract the position of the figure in each frame as row numeric data. This data is then classified as either a fall or a non-fall using trained binary classification models, including Support Vector Machines (SVM), Multi-Layer Perceptron (MLP), and Long Short-Term Memory (LSTM) networks. SVMs are effective in high-dimensional spaces due to their ability to find optimal hyperplanes that separate classes with the maximum margin. MLPs, with their multiple layers and activation functions, are well-suited for capturing complex patterns in data. LSTMs, designed to handle sequential data, excel in capturing temporal dependencies and long-range patterns due to their memory cells and gating mechanisms, which make them ideal for sequence prediction tasks.

Two benchmark fall datasets, Le2i and URFD (University of Rzeszow Fall Detection) (Charfi et al., 2013; Kwolek & Kepski, 2014), were used in the system's development. The Le2i Fall Detection Dataset was employed for training, offering a comprehensive collection of video recordings that simulate falls and non-fall activities under various conditions. The URFD dataset was utilised for testing. It includes video and sensor data collected during controlled experiments, where participants performed falls and other daily activities. The primary reason for using the URFD dataset was to leverage its sensor data registered per frame, providing a reliable ground truth for evaluation.

The integration of deep learning techniques has advanced the vision-based fall detection systems. Such systems can be categorised into auto-encoder based, CNN based, and LSTM based techniques. The input data for these systems can include RGB video, thermal images, and skeleton data (Alam et al., 2022). Skeleton data provides a powerful and efficient means of analysing human motion, making it a valuable tool for fall detection and other applications related to human activity recognition. It can be easily integrated into various machine learning models, including LSTM and CNN architectures, allowing these models to learn fall patterns based on joint movements. The use of skeleton data also facilitates real-time processing, enabling immediate detection and response to falls (Chua et al., 2015).

This project investigates the effectiveness of different models in fall detection by utilising pose estimation techniques and classification algorithms. Firstly, we employed MediaPipe and MoveNet for pose estimation. Secondly, we used SVM, MLP, and LSTM for classification, developing six predictive models in total. These models were trained with annotated data from the Le2i dataset and tested on fall sequences from the URFD dataset. Our findings demonstrate that the models effectively distinguished between falls and non-fall activities, with accuracy rates ranging from 78% to 84%.

# *Contribution log*

| | ***Total of 17 tasks completed over 57 days 222 hours in total*** | | | | | |
|---|---|---|---|---|---|---|
| | **Task title** | **Members** | **start date** | **end date** | **Hours spent** | **Contribution Percentage** |
| *1* | Find different techniques of detecting falls | Mohmmed Alsiraji | 08/06/2024 | 18/06/2024 | 33 H | 100% |
| *2* | Create dataset from features extracted by MoveNet | Mohmmed Alsiraji | 08/07/2024 | 16/07/2024 | 27 H | 100% |
| *3* | Create dataset from features extracted by MediaPipe | Mohmmed Alsiraji | 08/07/2024 | 15/07/2024 | 24 H | 100% |
| *4* | Build Mediapipe Function to suit our Task | Mohmmed Alsiraji | 19/06/2024 | 24/06/2024 | 18 H | 100% |
| *5* | Build MoveNet Function to suit our Task | Mohmmed Alsiraji | 19/06/2024 | 24/06/2024 | 18 H | 100% |
| *6* | Testing MOVENET and MEDIAPIPE for key points | Mohmmed Alsiraji | 29/06/2024 | 03/07/2024 | 15 H | 100% |
| *7* | Create and train MLP model | Mohmmed Alsiraji | 15/07/2024 | 19/07/2024 | 15 H | 100% |
| *8* | Find existing pose estimation models | Mohmmed Alsiraji Shah Sawar Jan | 11/06/2024 | 14/06/2024 | 12 H | 50%, 50% |
| *9* | Extraction of Key points from MOVENET and MEDIAPIPE | Mohmmed Alsiraji Shah Sawar Jan | 04/07/2024 | 07/07/2024 | 12 H | 50%, 50% |
| *10* | Prepare testing videos | Mohmmed Alsiraji Preety Batta | 08/07/2024 | 11/07/2024 | 12 H | 50%, 50% |

| | | | | | | |
|---|---|---|---|---|---|---|
| *11* | Evaluate MLP across MediaPipe and MoveNet | Mohmmed Alsiraji | 20/07/2024 | 23/07/2024 | 12 H | 100% |
| *12* | Create graphs of the evaluations | Mohmmed Alsiraji<br>Preety Batta | 23/07/2024 | 24/07/2024 | 6 H | 50%, 50% |
| *13* | Create ground truth testing | Mohammed Alsiraji<br>Shah Jawar Jan<br>Preety Batta<br>Afaq Ahmed Javed | 02/08/2024 | 03/08/2024 | 6 H | 25%, 25%, 25%, 25% |
| *14* | Create testing base-line for stick-models | Mohmmed Alsiraji | 12/06/2024 | 12/06/2024 | 3 H | 100% |
| *15* | Find how MoveNet works | Mohmmed Alsiraji | 15/06/2024 | 15/06/2024 | 3 H | 100% |
| *16* | Find out how Mediapipe works | Mohmmed Alsiraji | 16/06/2024 | 16/06/2024 | 3 H | 100% |
| *17* | Finalise the Pose Estimation Models Options | Mohmmed Alsiraji<br>Shah Sawar Jan | 18/06/2024 | 18/06/2024 | 3 H | 50%, 50% |

| The Detailed Description Table sorted by start date | | |
|---|---|---|
| **Task Name** | **Task Description** | **start date** |
| *Find different techniques of detecting falls* | I learned that falls can be detected by different approaches vision based and non-vision-based.<br>The first method uses image features from techniques like skeleton data, optical flow or appearance features such as colour and texture. Then analyse these features using advanced deep learning models.<br>Advantages are real-time processing and understanding of movement sequence leading to fall.<br>The second method mainly about using sensors to detect fall-based information such as acceleration. But the most of recent systems use combines the two methods to get the robustness of vision based and the accuracy of non-vision based. | 08/06/2024 |
| *Find existing pose estimation models* | This task is in the research milestone, we searched the web for pre-trained pose estimation model. The result was a list of 5 models:<br>1. Open Pose<br>2. Alpha Pose<br>3. MediaPipe<br>4. MoveNet<br>5. Detectron2 | 11/06/2024 |
| *Create testing base-line for stick-models* | What pose models use ground truth of testing the accuracy of the key-points. Can we create our own base-line to do tests accordingly?<br>Unfortunately this did not contribute to our work, as I found the models use metrics such as PDJ (Percentage of Detected Joints) or PCK (Percentage of Correct Keypoints) across multiple scenarios which would lead us out of the scope. | 12/06/2024 |
| *Find how MoveNet works* | Read an overview of the documentation from this page.<br>https://blog.tensorflow.org/2021/05/next-generation-pose-detection-with-movenet-and-tensorflowjs.html | 15/06/2024 |
| *Find out how Mediapipe works* | Read an overview of the documentation from this page.<br>https://ai.google.dev/edge/mediapipe/solutions/vision/pose_landmarker | 16/06/2024 |
| *Finalise the Pose Estimation Models Options* | We decided on which two models from the previous five are we going to continue working on and we choose: MediaPipe and MoveNet. | 18/06/2024 |
| *Build Mediapipe Function to suit our Task* | I read MediaPipe documentation and reviewed existing templates on how to use the model. Then, I created a function that is suitable to process an Image and annotate it with joints data. This task is important because if we analyse an image correctly, we can analyse an entire video.<br>https://blog.tensorflow.org/2021/05/next-generation-pose-detection-with-movenet-and-tensorflowjs.html | 19/06/2024 |

| | | |
|---|---|---|
| *Build MoveNet Function to suit our Task* | I read MoveNet documentation and reviewed existing templates on how to use the model. Then, I created a function that is suitable to process an Image and annotate it with joints data. This task is important because if we analyse an image correctly, we can analyse an entire video.<br><br>https://ai.google.dev/edge/mediapipe/solutions/vision/pose_landmarker | 19/06/2024 |
| *Testing MOVENET and MEDIAPIPE for key points* | I tested what keypoints MediaPipe detects there was 33 keypoints including the COCO 17 keypoint and tested what keypoints MoveNet detects there was the same 17 COCO keypoints. And most importantly I aligned the MediaPipe 33 keypoints by creating a dictionary to map new list according to MoveNet order since it has the less keypoints number. | 29/06/2024 |
| *Extraction of Key points from MOVENET and MEDIAPIPE* | This is where the focus shifts to having both models working on videos using the two image processing functions created in previous tasks.<br>To do:<br>1. The video must have for every frame 17 pairs of (x, y)<br>2. The 17 (x, y) must be annotated on every frame of the video.<br><br>We created a function that can take a single video and process each frame by MoveNet to give 17 joints (x, y). Like the MoveNet function we created a function that can take a single video and process each frame by MediaPipe to give 17 joints (x, y). The output for the two functions is similar, a CSV file of that video frames data including the joints data and an annotated video by each model. | 04/07/2024 |
| *Create dataset from features extracted by MediaPipe* | I implanted the function created in the extraction task on all the 40 videos, the output is 40 CSV files and 40 annotated videos. The CSV files we concatenated together thus creating a training dataset of MoveNet features for the classification algorithms. Most time spent on two parts; the execution time for all videos and inspecting we have the correct sequence. | 08/07/2024 |
| *Create dataset from features extracted by MoveNet* | I implanted the function created in the extraction task on all the 40 videos, the output is 40 CSV files and 40 annotated videos. The CSV files we concatenated together thus creating a training dataset of MediaPipe features for the classification algorithms. Most time spent on two parts; the execution time for all videos and inspecting we have the correct sequence. | 08/07/2024 |
| *Prepare testing videos* | The testing dataset URFD that we collected consists of 30 zip files each file has the sequence (frames) as png images. We prepared each file by reformatting it as an MP4 video using OpenCV. Output is 30 testing videos. | 08/07/2024 |
| *Create and train MLP model* | I was responsible for training MLP models on the two datasets created in earlier tasks, no train and test split were done. The models followed the same architecture, feedforward neural network (MLP) implemented using Keras, consists of a sequential architecture with a flatten layer followed by two fully connected (dense) layers of 128 and 64 neurons and a final dense layer with 1 neuron for binary classification output It was trained over 50 epochs. | 15/07/2024 |
| *Evaluate MLP across* | The MLP model was trained and ready to be used, I used it to predict the 30 testing videos. The same feature functions of MediaPipe and MoveNet | 20/07/2024 |

| | | |
|---|---|---|
| *MediaPipe and MoveNet* | were used with some adjustments on the annotation of the output videos and the CSV files to contain video number, frame index, predictions, frame execution duration, fall timestep per video, frame width and height. | |
| *Create graphs of the evaluations* | This task is about the evaluation of the six versions. The 3 evaluations done earlier gave us two CVS files each. We created 3 new tables from them by merging the pose models to the same classification algorithm used. 3 CSV files came out like {SVM, MLP and LSTM} and the interest columns in each are {predictions, execution time and detection time} for both MediaPipe and MoveNet.<br>With the interest columns we:<br>1. Plot the detection time.<br>2. Plot the Execution time. | 23/07/2024 |
| *Create ground truth testing* | After the final presentation, we felt the need for testing on some ground truth. The URFD dataset had a synchronised accelerator sensor data for all frames.<br>We a new list for each video a frame associated as the fall frame which has highest speed as the fall frame.<br>Then created a binary labels list matching the predictions length as they all on the same videos.<br>The output is a list of labels of the 30 videos according to sensor data.<br>The ground truth labels were tested against the six predictions lists. Then we plotted bar chart of the percentage. | 02/08/2024 |

# Personal Reflection

The project successfully utilised pose estimation models, specifically MediaPipe and MoveNet, to extract skeletal data from video footage. One of our main findings is the advantage of Mediapipe in execution time. These models were effective in detecting poses and, when combined with machine learning classifiers such as SVM, MLP, and LSTM, enabled differentiation between falls and non-fall activities. We trained and tested the models using benchmark datasets widely used in the academic field Le2i and URFD. We followed the recent approaches of solving the Fall detection problem. Although it is not a complete solution it is a great starting step.

To enhance the model, I suggest combining the features extracted by MediaPipe and MoveNet into a single dataset. MediaPipe provides consistent and reliable data by only returning results when confident in a detected pose, while MoveNet offers continuous data regardless of confidence. Therefore, merging these features could leverage MediaPipe's reliability and MoveNet's speed, resulting in a more robust model. This approach would shift the focus from comparing pose models to integrating them for improved prediction accuracy. Processing data in batches of frames, such as every 20 frames, could help the model recognise fall patterns more effectively. This could assist the LSTM model in creating sequential predictions based on groups of frames, reducing

false detections where activities resembling falls are incorrectly classified. Real-time detection could also improve the prototype, enabling practical, real-world applications. Real time detection also can improve this prototype, changing the perspective to analysing live stream, would put our work into real-world applications.

My background in video processing techniques was particularly useful, as I handled tasks related to video manipulation and processing, gaining extensive knowledge and skills in libraries such as OpenCV and HTML display. One of the key challenges I faced was learning how the pretrained pose models were working to ensure consistent keypoint extraction across models and balancing real-time processing with accuracy. We needed not only to know how to use the models but also to understand the inner workings of the models for effective handling volumes of video data. Understanding and adjusting pre-trained models were crucial for our objectives, though the time spent on specific models may not be applicable in other contexts.

The project highlighted the need for thorough data preprocessing and integrating diverse data sources. In future projects, I will prioritise the quality of detection and use a sequential approach to account for movement patterns, as deep learning models can learn from these patterns even if they are not immediately interpretable.

Collaboration with Shah Sawar Jan, Preety Batta, and Afaq Ahmed Javed was primarily through face-to-face meetings and text messages, planning our work, and dividing tasks according to each member's skills. Although we did not meet our clients, Dr. Jing Wang conveyed their requirements excellently. Ethical considerations, especially regarding privacy and data protection, are crucial when using video footage for fall detection. We used publicly available datasets to avoid these concerns but anonymising data and securing consent remain essential.

References

Alam, E., Sufian, A., Dutta, P., & Leo, M. (2022). Vision-based human fall detection systems using deep learning: A review. *Computers in Biology and Medicine, 146*, 105626. 10.1016/j.compbiomed.2022.105626

Charfi, I., Miteran, J., Dubois, J., Atri, M., & Tourki, R. (2013). Optimized spatio-temporal descriptors for real-time fall detection: comparison of support vector machine and Adaboost-based classification. *Journal of Electronic Imaging, 22*(4), 041106.

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

Espinosa, R., Ponce, H., Gutiérrez, S., Martínez-Villaseñor, L., Brieva, J., & Moya-Albor, E. (2019). A vision-based approach for fall detection using multiple cameras and convolutional neural networks: A case study using the UP-Fall detection dataset. *Computers in Biology and Medicine, 115*, 103520. 10.1016/j.compbiomed.2019.103520

Kwolek, B., & Kepski, M. (2014). Human fall detection on embedded platform using depth maps and wireless accelerometer. *Computer Methods and Programs in Biomedicine, 117*(3), 489-501.