PROJECT REPORT

on

PRE-FALL DETECTION BY OPENPOSE

A Thesis to be Submitted

in Partial Fulfilment of the Requirements

for the Degree of

Master of Technology

by

Subhajit Saha

Roll No : CS1933

under the supervision of

Dr. Ashish Ghosh

Professor and Head, Machine Intelligence Unit

Indian Statistical Institute

Kolkata-700108, India

CERTIFICATE

This is to certify that the dissertation entitled "**Pre-Fall Detection by OpenPose** " submitted by **Subhajit Saha** to Indian Statistical Institute, Kolkata, in partial fulfillment for the award of the degree of **Master of Technology in Computer Science** is a bonafide record of work carried out by him under my supervision and guidance. The dissertation has fulfilled all the requirements as per the regulations of the institute and, in my opinion, has reached the standard needed for submission.

A. Ghosh

Ashish Ghosh
Professor and Head,
Machine Intelligence Unit,
Indian Statistical Institute,
Kolkata-700108, India

Acknowledgement

I wish to express my sincere appreciation to my supervisor, Prof. Ashish Ghosh. I am delighted to have this project under his careful guidance. Special thanks go to the Machine Intelligence Unit (MIU) of ISI Kolkata for providing technical support and other essential resources. Senior research scholars specially Subhadip Boral, Debasrita Chakraborty, Anwesha Law gave me quality of time from their busy academic schedule so that I can complete this project conveniently. Last but not the least, I want to thank my parents, who constantly encouraged me in the study from my childhood and motivated me even in my bad days.

Subhajit Saha

Subhajit Saha
CS1933,
M.Tech in Computer Science,
Indian Statistical Institute,
Kolkata-700108, India

Abstract

Due to the improvement of medical science people worldwide are living longer. WHO has already predicted that the number of persons with age more than 60, will increase to 1.4 billion by 2030 and 2.1 billion by 2050. Where in 2019 it was 1 billion. Fall accidents have become one of the main health threats elderly. Here a pre-fall detection model based on OpenPose, a human posture estimation algorithm has been proposed to distinguish the normal daily activities and accidental falls. Four handcrafted features are extracted from the virtual skeleton returned by OpenPose and using those feature classification algorithms can classify falling and non-falling situations.

Keywords *Body Velocity; Change in vertical angle; Change in height; Variation of central line velocity; OpenPose; Video surveillance; Pre-Fall Detection*

Contents

1 Introduction

The vision-based video surveillance systems are used to increase safety against falls, fire, collisions, and explosions. Vision-based fall detection is a particular example of video surveillance which detects features in videos and uses them to detect falls. The fall alert detectors can measure when the user has suddenly fallen by detecting the abrupt changes in body posture and movement. But non-vision based fall-detection algorithm using an acceleration sensor node had proposed long ago. Despite the very promising result, in practice it is uncomfortable to wear a sensor every time; Especially an old person may forget to wear the sensor. So as an alternative vision -based models have achieved some popularity. But it is not an easy task because the final pose is usually a lying pose. In that case, the temporal motion from the previous posture of lying to the lying pose needs to be analyzed carefully. The proposed method should be robust enough to classify fuzzy but normal activities like jumping, crawling, dancing, etc as non-fall posture. Another important need of the model should be to get the prediction in minimum response time. The proposed model is also aimed to get higher prediction accuracy in real-time to provide the fallen person fastest medical treatment as well as to improve the quality of his/her elder life. To achieve fast response time, a real-time posture estimation algorithm called OpenPose has been used; For the classification job between fall and non-fall, four handcrafted; but easy to compute features are derived from the OpenPose. These only four features are proven good enough to get a very promising result. *Section 2* contains a brief discussion of some relevant research works. The base of our method i.e. OpenPose has described in *section 3* very precisely. *Section 4* provides details of the proposed idea. Derivation of the features and their justification to use here. In *Section 6* the result of the method in two datasets which are described in 5, has shown in support of the significance of the extracted features. At *section 7*, it is concluded by reviewing the success and limitations of the proposed method and further opportunities for improvements. *section 8* contains all the references of this project.

2 Related Work

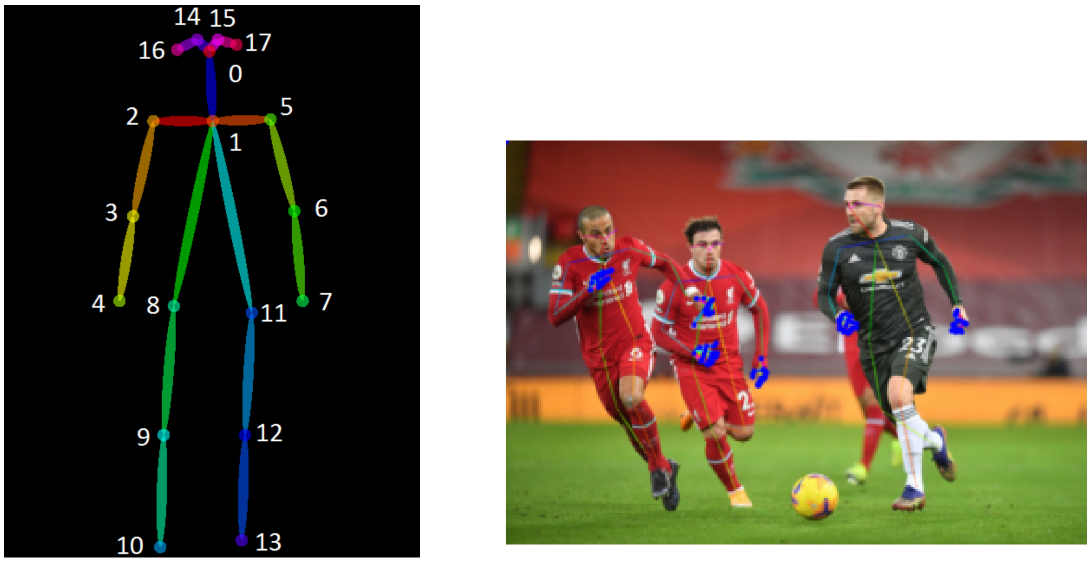
A novel fall-detection algorithm using an acceleration sensor node was presented by Dongha Lim et.al [8]. Deep-learning based Fall Detection algorithms for Embedded systems, Smartwatches, and IoT devices are discussed in detail over several data sets by Dimitri Kraft et.al [6]. Nuth Otanasap et.al [11] used multiple viewpoints provided multiple Kinect sensors using dynamic threshold based and center of gravity. Pre-fall detection alert would be triggered by the acceleration of head position compared with dynamic threshold-based approach and range of center of gravity compared with the base of the area. The inconvenience of these non-vision based models has discussed earlier.

LeilaPanahia and VahidGhods [12] proposed to estimate an ellipse around the body, and by obtaining its center distance from the floor they detected whether the body is on the floor or not. Then if it is on the floor for a long time it looks for the previous motion to detect the fall. But it is not a pre-fall detection and, if the person can move up quickly after a fall, it would return a false negative result. Many handcrafted feature-based models have been proposed by several researchers but AdriánNúñez-Marcos et.al [10] have successfully predicted fall or non fall by feeding CNN architecture by generating optical flow images.

There are some remarkable works done by Zhe Cao et.al [13] and [1] for that purpose. They presented a real-time approach to detecting the 2D pose of multiple people in an image, named as OpenPose. This algorithm predicts the posture of a person by returning his/her keypoints and edges denoting the joining limbs. In recent years there are several papers where the authors proposed to extract the feature from the output of OpenPose. Weiming Chen et. al [2] have created three hand-crafted features by OpenPose, declares a frame as falling if it exceeds all the three feature thresholds. Sungil Jeong et. al [5] have extracted SHCLC (Speed of Human Centerline Coordinate), and used LSTM to detect the falls. Several other works are using OpenPose also. A. Youssfi Alaoui et. al [14] tested conventional classification techniques like SVM, Random Forest, KNN and Decision Tree after extracting angle and distance between the keypoints (keyponts obtained by OpenPose) and they achieved very good result. Chuan-Bi Lin et. al [9] used the change in the 25 keypoints position as input feature and experimented for LSTM and GRU models. The algorithm proposed here also uses OpenPose as part of feature extraction.

3 OpenPose

OpenPose is a very fast real-time multi-person 2D posture estimation system to detect human body, hand, facial, and foot key-points from an input image. It was proposed by some researchers from Carnegie Mellon University. It returns 18,25 or 135 important body keypoint coordinates of each person of the input image with the confidence score of the respective keypoint. Also the runtime is invariant to number of person detected by OpenPose.



(a) (b)

Figure 1: (a) Index of 18 body keypoints (b) Hand and Body pose estimation

Index of the 18 keypoints from body posture estimation has shown precisely by Figure 1 and the following table.

KeyPoint Index					
Index	KeyPoint	Index	Keypoint	Index	Keypoint
0	Nose	6	Left Elbow	12	Left Knee
1	Neck	7	Left wrist	13	Left Ankle
2	Right Shoulder	8	Right Hip	14	Right Eye
3	Right Elbow	9	Right Knee	15	Left Eye
4	Right Wrist	10	Right Ankle	16	Right Ear
5	Left Shoulder	11	Left Hip	17	Left Ear

Table (1), Index with the corresponding KeyPoint returned by 18 keyPoint OpenPose

3.1 Two-branch Multi-stage CNN

Let, (w, h) be the shape of the input images. A feed forward network outputs a set of confedence score for the predicted keypoint set S or body part and another set of vector field L. Each of the vector field corresponds to a body limb. Body limb is the appropriate joining of two keypoints. Let $S = (S_1, S_2, ..., S_J)$ where $S_i \in R^{w \times h}$ is the set of predicted keypoints and $L = (L_1, L_2, ..., L_C)$ the set of vector fields with $L_j \in R^{w \times h \times 2}$.
The detailed description of L_j is given later. Nevertheless, two-branch multi-stage CNN architecture has used to generate keypoint set with confidence score S and that set of limb L. The predicted sets of keypoint and limb at one stage are sent to the next stage for prediction of both Keypoints and limbs, except for the initial stage. Mathematically,

$$S^t = \rho^t(F, S^{t-1}, L^{t-1}), \forall t > 1, S^1 = \rho^1(F) \qquad (1)$$

$$L^t = \phi^t(F, S^{t-1}, L^{t-1}), \forall t > 1, L^1 = \phi^1(F) \qquad (2)$$

where F is the original image feature and ρ^t and ρ^t are the first and second CNN function at time t respectively. F is generated by the first 10 layers of VGG-19.
The method is like supervised learning and L_2 loss function has used in training.

$$f_S^t = \sum_{j=1} \sum_p W(p) ||S_j^t(p) - S_j^*(p)||_2^2 \qquad (3)$$

$$f_L^t = \sum_{c=1}^{\mathbb{C}} \sum_p W(p) ||L_c^t(p) - L_c^*(p)||_2^2 \qquad (4)$$

Here S_j^* and L_c^* are the groundtruth values. $W(p) = 0$ if $p \in R^w \times h$ is not annotated else it is 1. Equation (3) corresponds to first and equation (4) corresponds to second CNN architecture. The ultimate goal is to minimize $\sum_{t=1}^{T}(f_S^t + f_L^t)$. Now both of the architectures need to be discussed separately.

3.2 Keypoint Detection

It is said earlier that the first architecture is devoted to find out set of keypoints with their corresponding confidence scores. It is calculated by a radial basis function. Which is,

$$S_{j,k}^* = exp(-\frac{||p - x_{j,k}||_2^2}{\sigma^2}) \qquad (5)$$

$x_{j,k}$ is the groundtruth position of jth keypoint for kth person. σ controls the spread of the rbf function. If the point $p \in R^2$ is very close to the groundtruth $x_{j,k}$ then that radial basis function gives very high score. So p is predicted as jth keypoint if $max_k S_{j,k}^*(p)$ is maximum among the other keypoints.

Before going to the most complicated and difficult part which is Keypoint Association; those vector fields of L which are named by the authors as *Part affinity Field i.e. PAF* need to be discussed.

3.3 Part affinity Field

It has introduced to keep the orientation and location of a limb. Naively speaking, it is just a vector filed of unit vectors from one keypoint to another keypoint . The Keypoint detection CNN returns J sets, for each jth set contains K predicted positions of jth keypoint. Where K is the number of detected persons in the input image. Now it is obvious that their is no association between any pair of keypoints that belongs to same keypoint set. Now consider a limb c joining two groundtruth keypoints, from $x_{j1,k}$ to $x_{j2,k}$. A point $p \in R^w$ belongs to that limb c if the vector $x_{j\vec{1,k}}p$ has non zero projection along the limb c and it's component along normal to the limb has less scaler value than the width of that limb. Mathematically, in that case the groundtruth PAF at p for limb c and kth person is,

$$L^*_{c,k}(p) = \frac{x_{j1,k}\vec{x}_{j2,k}}{||x_{j1,k}\vec{x}_{j2,k}||} \qquad (6)$$

else it is NUll vector.

The groundtruth value of PAF of p at c is calculated simply by doing the average of the scores using equation (6) over all K persons.

$$L^*_c(p) = \frac{1}{n_c(p)} \sum_k L^*_{c,k}(p) \qquad (7)$$

Here $n_c(p)$ denotes the number of nonzero vector at $p \in R^{w \times} h$

while testing, a confidence score is measured for a candidate limb joining d_{j1} and d_{j2} as

$$E = \sum_u L_c(p(u)) \frac{d_{j1}\vec{d}_{j2}}{||d_{j1}\vec{d}_{j2}||} \qquad (8)$$

To pick the points $p(u)$ line joining d_{j1} to d_{j2}, the convex combination of these two points are taken as the function $p : [0,1] \longrightarrow R^2$.i.e.

$$p(u) = u.d_{j1} + (1-u).d_{j2} \forall u \in [0,1] \quad (9)$$

u's are choosen in uniform distribution from $[0,1]$.

3.4 Multi-Person Parsing

PAF has been described in detail. The equation (10) has used to calculate the score of each candidate. The problem is to find out the true limb from J set of key points where each of them contains K same keypoint. This problem is known as K-dimensional matching problem. It is an NP-Hard problem. The researchers have simplified the problem to get quality matching. First, a minimal number of edges are chosen to make a spanning tree instead of computing from a complete bipartite graph. The second relaxation makes the already reduced problem more reduced by making many subproblems. Figure 2: (a) and the following graphs 2: (b) have summarized these steps properly considering only 3 keypoints and 2 persons.

Let $\phi_J = \{\{d_1^1, d_1^2, ..., d_1^{N1}\}, \{d_2^1, d_2^2, ..., d_2^{N1}\}, ..., \{d_J^1, d_J^2, ..., d_J^{NJ}\}\}$, be the set of sets of each keypoint.N_k represents the number of candidate person for kth keypoint, $\forall k \in \{1, 2, ..., J\}$. Obviously $d_j^i \in R^2$, the position of jth keypoint of ith person. Let, D_{j1}, $D_{j2} \in \phi_J$.A new variable x_{j1j2}^{mn} was defined, which is 1 if there is an edge d_{j1}^m to d_{j2}^n else 0. Mathematically each of the sub-problems can be written as a maximization problem,

$$max_{X_c} E_c = max_{X_c} \sum_{m \in D_{j1}} \sum_{n \in D_{j2}} E_{mn}.x_{j1j2}^{mn} \qquad (10)$$

s.t.

$$\forall m \in D_{j1}, \sum_{n \in D_{j2}} x_{j1j2}^{mn} < 2 \qquad (11)$$

$$\forall n \in D_{j2}, \sum_{m \in D_{j1}} x_{j1j2}^{mn} < 2 \qquad (12)$$

c is a limb type that is formed by the matching of keypoints from D_{j1} to D_{j2}. The optimal problem is to choose those d_{j1}^ms and d_{j2}^ns such that it would maximize E_c. The second CNN architecture would give a higher score to true joins and to maximize equation (10) those limbs would get added. With some obvious constraints as shown in equations (11) and (12). For any keypoint from D_{j1} there should be at most one matching keypoint in D_{j2}. If there is no matching then equation (11) returns 0, if there is any matching then the sum is 1. It can't be 2 or more as a human has every limb for once. like, no one has two or more left hands. Similar, interpretation can be made for equation (12) also. To get the optimal matching *Hungarian algorithm* [7] was used. After solving each subproblem for each limbs total PAF score E is counted of a framed, maximization of each subproblem would maximize E i.e.

$$\max_X E = \sum_{c=1}^{C} max_{X_c} E_c$$

.

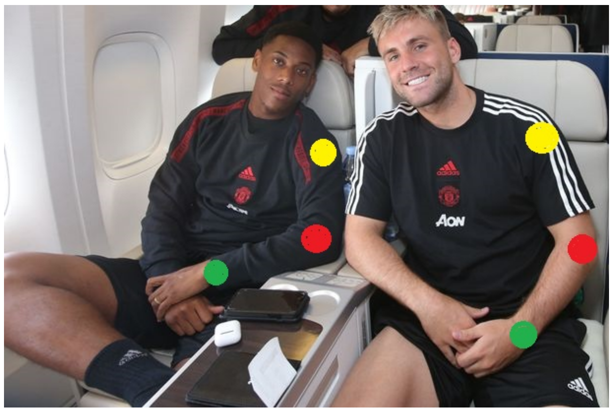


Figure 2: (a)

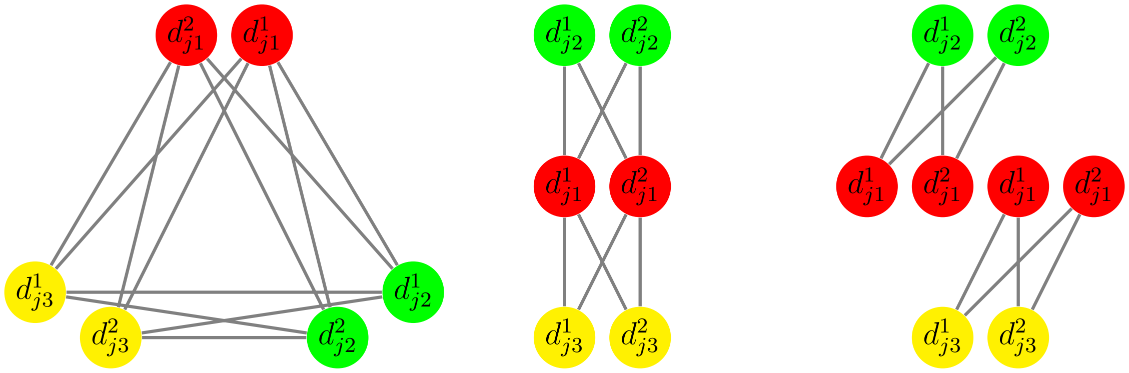


Figure 2: (b) Three steps for graph matching after body keypoint detection at Figure 2 (a), $K - Partite\ graph \longrightarrow$ Tree structure $\longrightarrow$ A set of Bipartite graph

4 Proposed Method

Openpose predicts the virtual skeleton in real time. Using the position of the keypoints here four different hand crafted features are proposed that carry out the predictive information for fall or non fall frames.

4.1 Change of vertical Angle

while a person is falling, in general, the body would not be vertically straight. But it would make some angle. This angle can be estimated by calculating the angle between the joining points of the neck with mid-hip provided by OpenPose and $(0, 1)$ direction. Ideally, if he/she is in a standing posture the body will make 0^o. And maximum if he/she is laying, which is the final position after each fall in most cases. But as said earlier, the temporal motion needs to be captured. So a window of length k has been used. That window length is a *hyper parameter* here. At a time t if θ^t is current angle, then we calculate $\theta^t - \theta^{t-k}$ as the feature value at time t of a video.

4.2 Change of Height

This feature is also intuitive as earlier. While falling the height of the body would gradually decrease. The height is computed by the calculating distance of the midpoint of left and right ankles from the nose. All the essential coordinates are generated by OpenPose as earlier. Similar to Vertical Angle, a window of length k has used. At a time t if h^t is current height, then we calculate $h^t - h^{t-k}$ as the feature value at time t of a video.

4.3 Body Velocity

As falling does happen very abruptly, due to sudden collision, imbalance or internal attack the body velocity should be higher than the usual situations. To estimate body velocity the mean of 14 keypoint coordinates (nose, neck, left and right shoulder, left and right knees, left and right elbows, left and right elbows, left and right ankles, left and right hips)are calculated for each frames. Similar to Vertical Angle, a window of length k has used. At a time t if μ^t is current mean body position, then we calculate $\mu^t - \mu^{t-k}$ as the velocity at time t of a video.

4.4 Variation of central line velocity

To capture the abrupt change in the body velocity another feature has been added here. For each frame coordinates of key points along the central line are calculated from OpenPose. The central line is nothing but the symmetric vertical line through the nose, neck, mid-hip, and mid-knee. As earlier a window k is taken and motion along with nose, neck, mid-hip, and mid-knee are computed. To capture the variation of these movements, a variance of these four points is calculated. This feature value would get higher while falling. For abbreviation, it is termed *VCLV* further in this writing.

Effectiveness of these four features can be visualized from the Figure 3,4. But none of these features are individually deterministic to predict fall or non-fall frames. But traditional supervised Machine learning models are tested here, they have shown some good results. So that window size k is taken same for all the features so that they can make feature vector together of dimension 4. The feature vector for each frame has been constructed but initially very few $k-1$ frames have no representation in the structured dataset. If due to noise or for any reason OpenPose does not predict the coordinate of a certain intermediate keypoint then Interpolation can be used to estimate those points. Here *linear interpolation* has used for this purpose.

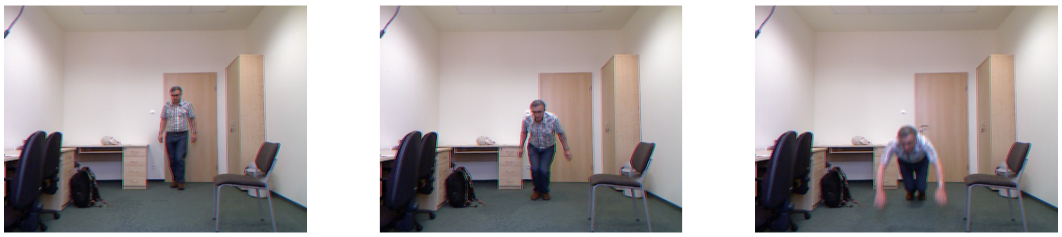
Complete Workflow

Input New Frame $\longrightarrow$ Generate Keypoints by OpenPose $\longrightarrow$ *Compute four features* $\longrightarrow$ *Use ML classifier* $\longrightarrow$ *Predict pre-fall or non pre-fall*

5 Datasets

5.1 UR Fall Detection Dataset

This is a publicly released RGB-D data set by the University of Rzeszow. This data set uses two Kinect cameras to record total 70 sequences, including 30 fall events and 40 daily activities. The two cameras are mounted in the ceiling and front door. The fall event, is mainly divided into two parts: one is falling on the way and the other is falling from a chair. Two Microsoft Kinect cameras are used to capture the video. Only the first camera data are used for the test purpose. Each frame has size $(480, 320)$.



(a) (b) (c)

Figure 3: (a) 6th , (b) 40th (c) 58th frame in a fall video from URFD 25th fall video

5.2 MMU Fall Detection Dataset

This dataset was made and used by Jia Luen Chua et.al [3] at Multimedia University, Malaysia. The video data were captured from an uncalibrated IP camera. The picture resolution is $(320, 240$. The dataset simulates 21 falls with various postures like forwarding fall, backward fall, side fall. It has also 30 daily activities data like walking, sitting down, squatting down, etc. All the videos were recorded inside a single room. The camera was installed at a higher height than the first camera used in URFDD. A promising result invariance to the dataset will assure the honesty and effectiveness of the proposed method.

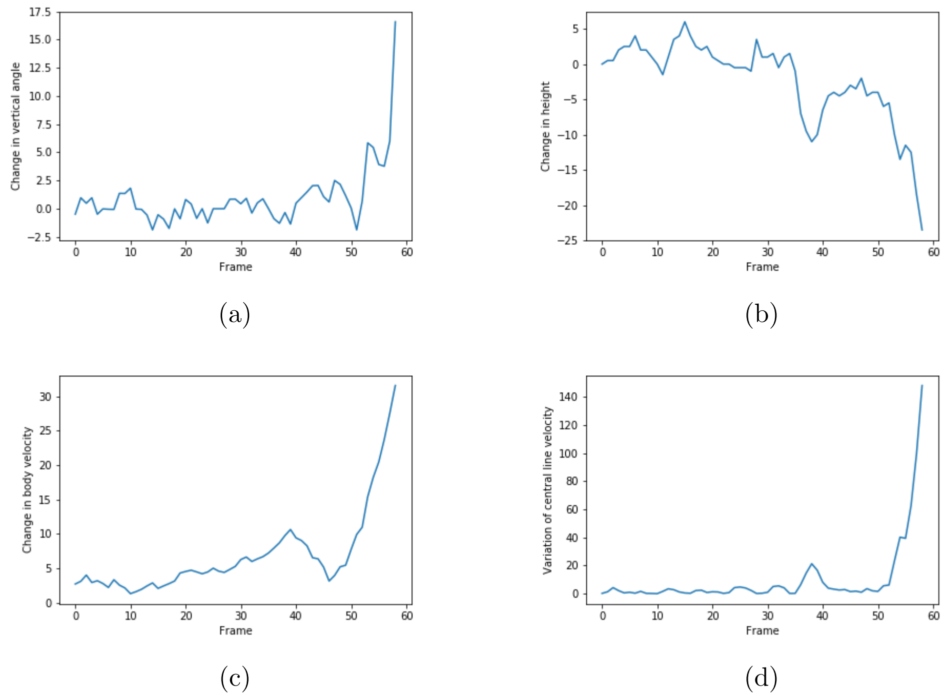


Figure 4: Four feature values over time frame in URFDD 25th fall video; change of angle (a), change in height (b), body velocity (c), variation of central line velocity (d)

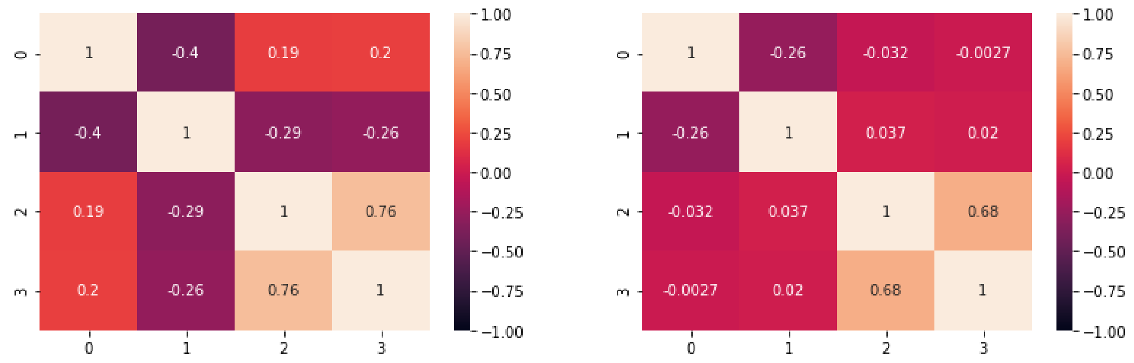
6 Experimental Results

The proposed method was tested on these two datasets. There are some necessary terminologies which are used here.
True positive(TP) No of predicting a falling frame to truly falling frame.
True negative(TN) No of predicting a non falling frame to truly non falling frame.
False positive(FP) No of Predicting a falling frame to truly non falling frame.
False negative(FN) No of Predicting a non falling frame to truly falling frame.

All the frames of both datasets contain one volunteer performing fall or non fall activities. An empty room would not return any key point by OpenPose. only first 14 keypoints are needed (excluding ears and eyes) for the proposed method. But due to noise or view problem of the camera some keypoints may be missed by OpenPose. If the position of the nose is not predicted and the eyes pair are predicted, then the nose is approximated by the mid point of eyes. If eyes are not predicted then it may again approximated by the mid points of ears. In general if an intermediate point is not predicted by OpenPose then it has approximated by interpolation techniques. Here linear interpolation has used to predict those intermediate keypoints.

For training purpose 20 daily activities and 15 fall videos were selected of URFDD. The remaining videos are used in testing. The window length k has taken 3 for the best outcome in the test data. For the classification step it has been tested over several well-known classification techniques.Like MLP, SVM, KNN , Randomm Forest etc. but *Random Forest*, has shown very promising results. The Random Forest used here contains 100 trees, each tree with maximum depth 2 and *Gini impurity* for an optimum split of the trees. A falling frame was labeled earlier as 0 and a non-falling frame as 1. So the target is 0 or 1. Nevertheless, boosting algorithms also do not work that well compared to other algorithms. To evaluate the performance of the proposed method four well-known evaluation metrics have been used. They are defined as **Accuracy** $= \frac{TP+TN}{TP+TN+FP+FN}$, **Precision** $= \frac{TP}{TP+FP}$, **Recall** $= \frac{TP}{TP+FN}$ and **F1 score** $= \frac{2*Precision*Recall}{Precision+Recall}$.
The second dataset was splited into nearly 1 : 1 ratio for train-test purpose. The hyperparameter window size was chosen 2. Those classifier used in URFDD are also trained and tested here. The outcome is similar in both these two datasets. Again Random Forest classifier has outperformed other models. All the tests are performed in python3 by using a local machine of the feature i7-3520M, 2.90GHz with 16GB RAM.

To view the relationship among these features, correlation matrix has computed for each dataset. since the features may not follow normal distribution instead of *Pearson Correlation*, *Spearman correlation* has computed. Figure 5 shows it clearly, the $(i, j)th$ term of each matrix shows the correlation between ith and jth feature. Value close to -1 or $+1$ represents strong correlation. similar. In that case, one feature is nothing but a linear transformation of another feature. These kinds of similar features do not improve some machine learning models that much as there is no additional information by adding a new similar feature. On the other hand for a time-constrained model, more dimensions may make the prediction time big. As one of the goals of this model is to make predictions fast, it is ideal to have low dimension data. Figure 5 shows that there is no feature pair with a correlation close to $1(> 0.8)$ in both datasets. Body Velocity and VCLV have shown a good positive correlation, which is obvious also zero body velocity implies zero VCLV. Another common outcome in both datasets is that the first two features are negatively correlated. It is also intuitive. If the motion is bending while falling or sitting if the change in height is negative then the change in vertical angle will increase.



(a) (b)

Figure 5: (a) Correlation matrix of features from training data of URFDD (b) Correlation matrix of features from training data of MMUFDD

In Figure 6 continuous video frames starting with non-fall to after fall state is shown. The next table shows the feature values with random forest classifier prediction of those frames. Predicting falling state is said as Yes in red, else said as No in green. It would also visualize how each feature value changes as the motion changes. Falling starts at (l)th frame and ends at (q)th frame. The random forest classifier also gives the same prediction. The first feature value increases, the second feature value decreases, the third feature value increases and the last feature increases rapidly while falling. This again justifies the usefulness of the hand-crafted features.

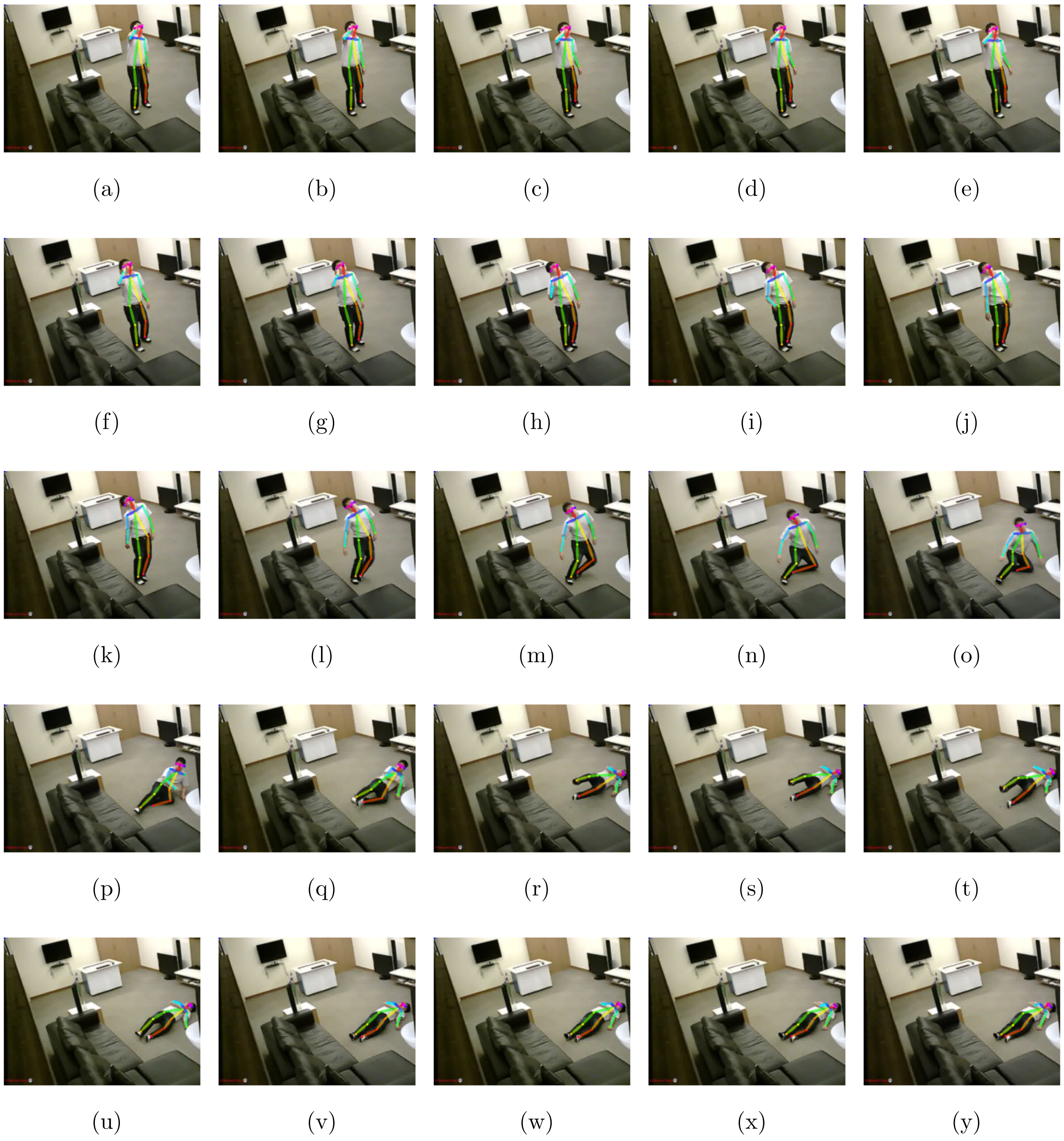


Figure 6: A continuous video frames of MMUFDD after constructing virtual skeletons from non fall state to after fall state. All the frames are indexed by (a) to (y)

Frame	Change of Vertical Angle	Change Of Height	Body Velocity	VCLV	Prediction
(a)	0.62	-1.5	2.78	1.75	No
(b)	4.02	-5	5.44	6.77	No
(c)	3.69	-4	8.14	10.10	No
(d)	5.92	-0.5	15.42	25.48	No
(e)	7.55	3.5	14.82	17.38	No
(f)	5.39	0.5	8.57	4.38	No
(g)	4.74	-2.5	12.95	6.51	No
(h)	0.47	1.5	12.5	20.91	No
(i)	-0.14	-2.5	12.95	6.51	No
(j)	0.65	-6	22.43	56.45	No
(k)	-3.40	-16.5	35.30	131.824	No
(l)	-9.68	-26.5	39.30	221.559	Yes
(m)	-0.63	-32.5	48.46	383.367	Yes
(n)	31.57	-29	56.44	1048.61	Yes
(o)	34.32	-19.5	38.69	419.233	Yes
(p)	35.09	-36	28.70	67.5345	Yes
(q)	23.46	-22.5	19.45	69.94	Yes
(r)	-5.57	11.5	3.86	10.89	No
(s)	-3.22	11	17.45	65.26	No
(t)	-3.70	5.5	16.88	122.114	No
(u)	-2.68	5.5	5.31	14.80	No
(v)	1.35	-1	2.63	0.14	No
(w)	0.44	-4	1.66	2.08	No
(x)	-1.41	-1.5	2.57	0.56	No
(y)	-2.68	5	0.68	0.48	No

Table (2): Feature scores of the frames from Figure 6 and their corresponding predictions

Results by Random Forest Classification				
Dataset	Accuracy	Precision	Recall	F1 Score
URFDD	0.95	0.69	0.71	0.70
MMUDD	0.99	0.78	0.74	0.76

Table (3) Results by RF classification

The above table is the summary of the experimental results. The results are promising but also there are more accurate results by the other researchers. Nevertheless, the falling on the way towards the camera does not capture the true vertical angle. Those falling contribute false-negative output. Also, some fuzzy movements like move up after fall, etc give false positive output. But for practical purposes, we can tune the classification threshold to make fewer false positives and allow some false negatives. While falling if there are 10 such frames only one true positive is enough to provide the needy person some help. For example, If we would tune the classifier threshold for the URFDD dataset, it predicts at least one true falling frame in all fall videos and no falling frame in all daily activity videos. So practically this extracting feature based model using OpenPose is very efficient for video surveillance.

7 Conclusion

This OpenPose based feature extracting method has been shown to work well in a complex space environment with lower equipment costs. Still, the result is not that good compared to the other models (The overview of current vision-based fall detection methods and their performances has been documented in [4] precisely). By adding other features or other advanced classifiers may improve the proposed methodology. However, the joint points will be lost in some postures and actions, causing the model to have coagulation during training. Interpolation is applicable only for intermediate missing data, extrapolation based techniques to estimate the key points at the start and end of the video are not that good. In some cases few frames are not taking as input data, also the handcrafted feature needs all such key points to generate features. One possible solution can be the use of multiple cameras from different positions to get many views of the person. But it is not an optimized idea financially. Similar algorithm with less need of keypoints may solve this issue. So, there is some bright scope of improvement from the proposed OpenPose based feature extracting method.

References

1. Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(1):172–186, 2021.
2. Weiming Chen, Zijie Jiang, Hailin Guo, and Xiaoyang Ni. Fall detection based on key points of human-skeleton using openpose. *Symmetry*, 12(5), 2020.
3. Jia-Luen Chua, Yoong Choon Chang, and Wee Keong Lim. A simple vision-based fall detection technique for indoor video surveillance. *Signal, Image and Video Processing*, 9(3):623–633, 2015.
4. Jesús Gutiérrez, Víctor Rodríguez, and Sergio Martin. Comprehensive review of vision-based fall detection systems. *Sensors*, 21(3), 2021.
5. Sungil Jeong, Sungjoo Kang, and Ingeol Chun. Human-skeleton based fall-detection method using lstm for manufacturing industries. In *2019 34th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC)*, pages 1–4, 2019.
6. Dimitri Kraft, Karthik Srinivasan, and Gerald Bieber. Deep learning based fall detection algorithms for embedded systems, smartwatches, and iot devices using accelerometers. *Technologies*, 8(4), 2020.
7. Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.
8. Dongha Lim, Chulho Park, Nam Kim, Sang-Hoon Kim, and Yun Seop Yu. Fall-detection algorithm using 3-axis acceleration: Combination with simple threshold and hidden markov model. *Journal of Applied Mathematics*, 2014, 09 2014.
9. Chuan-Bi Lin, Ziqian Dong, Wei-Kai Kuan, and Yung-Fa Huang. A framework for fall detection based on openpose skeleton and lstm/gru models. *Applied Sciences*, 11(1), 2021.
10. Adrián Núñez-Marcos, Gorka Azkune, and Ignacio Arganda-Carreras. Vision-based fall detection with convolutional neural networks. *Wireless Communications and Mobile Computing*, 2017:1–16, 12 2017.
11. Nuth Otanasap and Poonpong Boonbrahm. Pre-impact fall detection approach using dynamic threshold based and center of gravity in multiple kinect viewpoints. In *2017 14th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pages 1–6, 2017.
12. Leila Panahi and Vahid Ghods. Human fall detection using machine vision techniques on rgb–d images. *Biomedical Signal Processing and Control*, 44:146–153, 2018.
13. Sen Qiao, Yilin Wang, and Jian Li. Real-time human gesture grading based on openpose. In *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*.
14. Abdessamad Youssfi Alaoui, Sanaa El Fkihi, and Oulad haj thami Rachid. Fall detection for elderly people using the variation of key points of human skeleton. *IEEE Access*, PP:1–1, 10 2019.