Computers in Biology and Medicine 146 (2022) 105626

Contents lists available at ScienceDirect

Computers in Biology and Medicine

journal homepage: www.elsevier.com/locate/compbiomed

Vision-based human fall detection systems using deep learning: A review

Ekram Alam [a], Abu Sufian [b], Paramartha Dutta [c], Marco Leo [d,*]

[a] Department of Computer Science, Gour Mahavidyalaya, West Bengal, India
[b] Department of Computer Science, University of Gour Banga, India
[c] Department of Computer and System Sciences, Visva-Bharati University, India
[d] National Research Council of Italy, Institute of Applied Sciences and Intelligent Systems, 73100, Lecce, Italy

ARTICLE INFO

Keywords:
Human Fall Detection
Fall Detection Metrics
Sensitivity
Specificity
Accuracy
Human Fall Datasets
Multiple Camera Fall Dataset
Le2i Fall Detection Dataset
URFD

ABSTRACT

Human fall is one of the very critical health issues, especially for elders and disabled people living alone. The number of elder populations is increasing steadily worldwide. Therefore, human fall detection is becoming an effective technique for assistive living for those people. For assistive living, deep learning and computer vision have been used largely. In this review article, we discuss deep learning (DL)-based state-of-the-arts non-intrusive (vision-based) fall detection techniques. We also present a survey on fall detection benchmark datasets. For a clear understanding, we briefly discuss different metrics which are used to evaluate the performance of the fall detection systems. This article also gives a future direction on vision-based human fall detection techniques.

1. Introduction

According to a report of the United Nations (UN) [1], the elderly population of age 65 years or above was 727 million globally in 2020, which is expected to reach 1.5 billion (more than two times) by 2050. The percentage of the elderly population of this age group was 9.3% in 2020 which will be 16.0% in 2050. Due to a better standard of life and improvement in healthcare, the average life expectancy is increasing. If this trend continues, Very soon there will be more elderly populations than adults. So, the caring of the elders may become a big problem due to the scarcity of caretakers. These days we are very much dependent on technologies, and for the elderly population, this is not an exception. One of the main health problems in the elderly population is fall due to weakness and other reasons. Falls are one of the most common reasons for hospitalization for elderly people [2]. Not only prevention but detection of falls as early as possible is very crucial for the health of the concerned person. A slight delay in detecting a fall and providing medical help can be fatal. The chance of dying a person within 6 months after the fall is 50% if the person is on the floor for more than 1 h after the fall [3,4]. So, detection, prediction, and raising alarm to take action as early as possible is very important. Automatic collection and reporting of fall incidents can be used to analyze the cause of falls and can help to prevent falls. Nowadays telehealth is becoming very effective for frequent outbreaks of pandemics and epidemics [5,6]. Fall detection can be done using wearable sensors; like gyroscopes and accelerometers; and vision sensors like RGB cameras, infrared cameras, depth cameras, and 3D-based methods using camera arrays. Wearable devices capture abnormal values like velocities and angles from the sensors and raise an alarm to alert the user and third party. One of the main problems in wearable devices is the requirement of frequent charging. For older adults, it may be difficult to continuously monitor the battery charging status and charge it frequently. Wearable sensors are also not so comfortable and have many side effects. Due to frequent charging, uncomfortable and other side effects of the sensors this option is not so suitable for elderly people. So, non-intrusive vision-based sensors can be a good choice [7]. Vision-based fall detection is a better alternative that provides a low-cost solution for the fall detection problem [8]. Modern artificial intelligence, specifically DL [9] is very effective for this kind of task [10,11]. Also, due to the increased use of IoT [12] solutions and the more uses of cameras in airports, bus stands, railway stations, roads, streets, and homes, vision-based methods for fall detection is a good choice for the future as well. Though there are many other approaches for fall detection, DL based approach is gaining momentum. As compared to other methods, there is no need of handcrafted feature extraction in DL. Automatic feature extraction is possible in DL-based methods. DL is also popular due to its generalization. A model trained

* Corresponding author.
E-mail addresses: ealam4u@gmail.com (E. Alam), sufian.csa@gmail.com (A. Sufian), paramartha.dutta@gmail.com (P. Dutta), marco.leo@cnr.it (M. Leo).

https://doi.org/10.1016/j.compbiomed.2022.105626
Received 2 January 2022; Received in revised form 8 March 2022; Accepted 6 April 2022
Available online 27 May 2022

on a dataset can be used for a different problem using transfer learning. The performances of DL-based techniques are also very high as compared to other methods. DL can also be used in low computing edge devices using transfer learning and few-shot learning. Fall can be of different types and duration. According to Bendary et al. [13], fall can be of three types namely forward, backward, and sideway. Putra et al. [14] classifies fall detection into forward (ff), backward (fb), right-side (fr), left-side (fl), blinded-forward (bff), and blinded-backward (fb). Fig. 1 shows this classification.

The rest of the paper is organized as follows. Section 2 mentions some related works. Section 3 discusses metrics to evaluate the performance of the different fall detection techniques. A brief description of the publicly available human fall dataset (HFDS) is provided in section 4. Section 5 reviews the vision-based fall detection techniques using deep learning. Finally section 7 conclude with some future direction.

2. Related works

Gutierrez et al. [15] published a recent review work on vision-based fall detection methods. This paper reviewed all types of vision-based fall detection algorithms. Our paper specifically discusses only deep learning-based non-intrusive (vision) fall detection methods in detail. Also, we have classified our review based on different types of deep learning models. Wang et al. [16] and Xefteris et al. [17] provided a multi-model review paper where the signals from different sensors (wearable sensors, vision sensors. ambient sensors) were fused. Our paper only reviews vision sensors. Rastogi et al. [18] presented a brief machine learning-based fall detection review paper which considers all types of sensors (vision, wearable, ambient). Our paper describes only deep learning (a subset of machine learning) based fall detection techniques considering only vision sensors. In this paper, we have reviewed vision-based non-intrusive human fall detection (HFD) techniques using deep learning.

3. Evaluation Metrics

A fall detection problem can be considered a binary problem. In a binary problem, we have two outcomes with respect to each input, True (T) and False (F). For fall detection problem T means occurring a fall and F means no fall. Many works were done to detect the falls. To compare and evaluate the performances of the experiments done by different researchers, many metrics have been designed. Some of the important metrics used by most of the researchers are given below [19].

• **Sensitivity**	• **Specificity**	• **Accuracy**
• **F Score**	• **Precision**	• **Geometric Mean**

All of the above-mentioned metrics can be defined on the terms of TP (True Positive), TN (True Negative), FP (False Positive), and FN (False Negative). TP means detecting falls correctly. FP denotes detecting activities of daily living (ADL) as a fall. FP means wrongly detecting a normal activity as a fall. Similarly, FN means detecting a normal activity as a fall.

Sensitivity (Recall) is the ratio between TP and sum of TP and FN as shown in equation (1). It is the proportion of true positive falls that are correctly identified by a fall detection system. Sensitivity signifies that how correctly an experiment can detect an actual fall as fall.

$$Sensitivity = \frac{TP}{(TP + FN)} \tag{1}$$

Similarly, **Specificity** is the ratio between TN falls and sum of TN falls and FP falls as shown in equation (2). It signifies that how correctly an experiment can detect a normal activity as normal activity.

$$Specificity = \frac{TN}{(TN + FP)} \tag{2}$$

Sensitivity and Specificity do not give the full information about the accuracy of an experiment. To know how accurate an experiment is, both sensitivity and specificity are important. High sensitivity and low specificity and high specificity and low sensitivity are not so accurate. For, good accuracy both sensitivity and specificity should be high. **Accuracy** combines both sensitivity and specificity as shown in equation (3).

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{3}$$

Similarly, **Precision** can be defined mathematically as shown in equation (4). It is the ratio of TP falls to all positive falls (TP and FP).

$$Precision = \frac{TP}{(TP + FP)} \tag{4}$$

F Score (FS), also known as the F1 score is the harmonic mean of precision and sensitivity. Mathematically, it can be defined as shown in equation (5). FS combines the features of precision and recall.

$$FS = \frac{2}{\left(\frac{1}{recall} \times \frac{1}{precision}\right)} = 2 \times \frac{precision \times recall}{precision + recall} \tag{5}$$

Some researchers have also used **Geometric Mean (GM)** [20,21] of sensitivity and specificity for the analysis of their experiments. The GM can be calculated using the following equation.

$$GM = \sqrt{(SE * SP)} \tag{6}$$

4. Benchmark public human fall datasets

DL-based techniques are generally very data-hungry. It requires lots of data for training, testing, and validation. Benchmark HFDS can be utilized for this purpose. Benchmark HFDSs are also required for the evaluation and comparison of different HFD techniques. The publicly available benchmark HFDSs [21–27] have been created by performing fall and ADL activities by some subjects. Many researchers also used some pre-trained DL models. These models were pre-trained on other human activity recognition (HAR) datasets like KTH [28], UCF-101 [29], MSRDailyActivity3D [30] etc. and other large size datasets like ImageNet [31], Microsoft (MS) COCO [32], NTU RGB + D 120 [33]. Table 2 summaries the details of HFDSs. The full forms of the acronyms

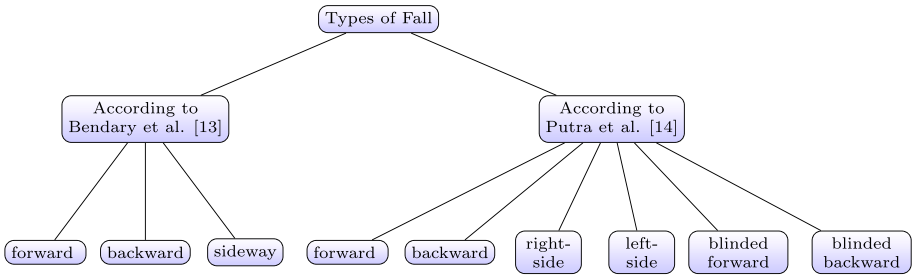


Fig. 1. Classification of falls by Refs. [13,14].

Table 1
The full forms of the acronyms used in Table 2.

CN →	No. of Cameras	SN →	No. of Subjects
MP →	Multi-persons	Occ. →	Occlusion
FAN →	No. of Fall Activities	AN →	No. of ADL Activities
TAN →	Total No. of Activities	FDN →	No. of Fall Data Sequences
ADN →	No. of ADL Data Sequences	TD →	Total No of Data Sequences

used in Table 2 are given in Table 1. A brief description of the HFDSs are given below.

4.1. Multiple cameras fall dataset (MCFD)

Auvinet et al. [22] introduced a dataset using 8 IP cameras which contains total 24 activities. Each activity was recorded using 8 different cameras from different viewpoints simultaneously. The recordings are available in AVI format. The fall and ADL activities were performed by a single subject. Falling and other activities recorded in this dataset are given below.

1. Walking	2. Falling	3. Lying on the ground	4. Crouching
5. Moving down	6. Moving up	7. Sitting	8. Lying on a sofa
9. Moving horizontally	10. Standing up		

4.2. Le2i fall detection dataset (Le2i FDD)

Charfi et al. [21] presented fall detection dataset [21] using a single RGB camera. Originally they named it S, now commonly known as the Le2i FDD dataset. Nine subjects performed 3 different types of fall activities (forward falls, balance loss, falls from sitting) and 6 different ADL activities (sitting, walking, standing, moving a chair, housekeeping) and captured 143 fall videos and 79 ADL videos. Videos were captured in four different environments (Home, Coffee room, Office, Lecture Room). Activities were performed in varying the various factors like light, clothes, the color of clothes, texture of clothes, shadows, reflections, camera view, etc.

4.3. University of rzeszow fall detection (URFD) dataset

URFD dataset [23] was presented by Kwolek and Kepski in 2014. This dataset contains 40 ADL sequences and 30 fall sequences. Two MS Kinect cameras were used for capturing the depth and RGB images. This dataset also contains accelerometer data.

4.4. SDUFall

Ma et al. [24] introduced a depth fall dataset named SDUFall [24] using an MS Kinect camera. Ten subjects (male and female) performed the following six actions.

1. Falling down	2. Walking	3. Bending
4. Lying	5. Squatting	6. Sitting

Each subject repeated an action 30 times with variations in light, room layout, camera angle and position, carrying an object, etc. Total 1800 (6 × 30 x 10) video clips of 8-s duration were recorded using an MS Kinect camera installed at a height of 1.5 m. The files were saved in AVI format.

4.5. High quality fall simulation dataset (HQFSD)

Baldewijns et al. [34] presented the HQFSD [34] using five web cameras (640 × 480 resolution, 12 fps) recorded in nursing home environments. Ten subjects performed 55 fall scenarios of average length 2:45 min (min length 50 s, max length 4:58 min). The total duration of fall recording by each camera is 2:25:54 h. Fall scenarios differ in the use of walking aid (walker, wheelchair, no walking aid), fall speed (slow, fast), moving objects during the fall (blanket, walker, wheelchair, chair, none), starting pose (standing, sitting, squatting, bending over, transitions), ending positions (lying on the floor, getting back up after fall), etc. Total 17 different ADL scenarios were performed of average length 20:39 min (min length 11:38 min, max length 35:30 min). The total duration of ADL recording by each camera is 5:50:29 h. The following 13 ADL scenarios were recorded. The videos were saved in AVI format.

1. Sitting	2. Eating and drinking	3. Picking something from the floor
4. Transition	5. Making the bed	6. Putting and removing shoes
7. Sleeping	8. Changing clothes	9. Coughing and sneezing
10. Reading	11. Walking	12. Getting into and out of bed
13. Wheelchair to chair and vice versa		

4.6. Thermal simulated fall dataset (TSFD)

Vadivelu et al. [35] presented a TSFD [35] using a single FLIR ONE thermal camera. The camera was mounted on an Android phone. Fall activities were done in a room environment. TSFD contains 9 ADL video segments (VS) and 35 fall VS.

4.7. SisFall

Sucerquia et al. [25] presented a fall dataset named SisFall [25]. This dataset was basically created using two accelerometers and a gyroscope. SisFall also contains video sequences [36]. It has 19 ADL (Walking slowly, Stumble while walking, Jogging slowly et.) video sequences and 15 (forward, lateral, vertical, etc.) fall video sequences.

Table 2
Benchmark human fall datasets.

Dataset	Year	Sensor Type	Camera Type	CN	SN	MP	Occ.	FAN	AN	TAN	FDN	ADN	TD	Link
MCFD [22]	2010	Vision	IP Cameras	8	1	No	Yes	–	–	–	–	–	24 × 8	Link
Le2i FDD [21]	2013	Vision	RGB	1	9	No	Yes	3	6	9	143	79	222 VS	Link
URFD [23]	2014	Vision	Depth, RGB	2	1	No	No	–	–	–	30 Seq	40 Seq	70	Link
SDUFall [24]	2014	Vision	Depth	1	10	–	–	1	5	6	–	–	1800 videos	Link
HQFSD [34]	2016	Vision	Web Camera	5	10	Yes	Yes	24	13	37	24 120 VS	65 VS	185 VS	Link
TSFD [35]	2016	Vision	Thermal	1	–	–	–	–	–	–	35 VS	9 VS	44	Link
SisFall [25]	2017	Wearable, Vision	RGB	–	15	–	–	15	19	34	15	19	34	Link
UP-Fall Detection [26]	2019	Wearable, Ambient and Vision	RGB (PNG)	2	17	No	No	5	6	11	–	–	–	Link
FPDS [27]	2019	Vision	RGB	1		Yes	–	–	–	–	1072	1262	2062 img	Link

4.8. UP-fall detection dataset

Martinez et al. [26] presented a multi-modal UP-Fall Detection dataset using wearable, ambient, and vision sensors. Seventeen subjects of different ages (18–24), genders (9 males and 8 females), and weights (mean 66.8 kg) performed 11 different activities (6 ADL, 5 Fall). Each activity was repeated three times. The ADL activities performed are given below.

1. Walking	2. Standing	3. Jumping
4. Sitting	5. Laying	6. Picking up an object

The Fall activities performed are given below.

1. Falling backward	2. Falling forward using hands	3. Falling forward using knees
4. Falling sideward	5. Falling sitting in an empty chair	

4.9. Fallen person dataset (FPDS)

Maldona et al. [27] introduced a fall dataset named FPDS [27] using a single camera fitted in a robot at a height of 76 cm. FPDS contains 1072 falls and 1262 ADL (Standing, Walking, Lying, Sitting, etc) manually labeled images. More than one subject can be in a single image. The height of subjects was from 1.2 m to 1.8 m. The activities were done in 8 different environments with varying conditions of light, shadows, reflections, etc.

5. Review on recent state of the art

Though research on vision-based fall detection using Neural Network has started from the year 2012 [37], the first vision-based fall detection work using deep learning is done by Feng et al. [38] in the year 2014. In this paper, we have reviewed the DL-based non-intrusive (vision) HFD methods since 2014. We have selected the different papers from Web of Science, Scopus, Google Scholar using different permutations of searching keywords. We have divided the keywords into two parts. In first part we used the keywords related to falls(like “video fall”, “fall accident”, “motion fall” etc). In second part we used the deep learning related keyword (like “deep learning”, “CNN”, “auto-encoder”, “LSTM” etc). We separated the keywords of 1st part and 2nd part using ‘OR’. We inserted ‘AND’ between the 1st part and 2nd part keywords to generate different permutations. We filtered the generated papers manually to fit into the scope of this work. This is shown in Fig. 2.

We have classified the HFD techniques based on the DL model as given below.

- Convolutional Neural Network (CNN)
- Auto-encoder
- Hybrid
- Long Short Term Memory (LSTM)
- Multi Layer Perceptron (MLP)

Three tables (Basic Information, Evaluation Metrics, Optimization Details) for each type of DL model have been created. The basic steps for HFD is shown in Fig. 3.

5.1. CNN based techniques

CNN is the one of most used DL models in computer vision and HFD is not the exception. Different sub-types of CNN are 3D CNN, TDCNN (time delay convolutional neural network), GCN (graph convolutional network), etc. The details about the experiments reviewed in this section are summarized in Tables 7–9.

From the year 2016, vision-based fall detection using deep learning especially CNN gained momentum. This year, a work was reported by Doulamis [39] using time delay neural network (TDNN) [40]. At first, the human shape was extracted from the background, and then object tracking was applied. Here an adaptable deep learning approach was used for human detection and classification purpose. Using object tracking, an object trajectory was generated over each frame of the video. The generated object trajectory was geometrically analyzed to compute different 3D measurements to detect a human fall. Finally, A sequence of geometric features from the detected human object was fed to a TDNN to detect the human falls. A single monocular camera was used.

Fan et al. (2017) [41] introduced fall detection method using dynamic images [42]. This method not only detects the falls but also tells the duration of the fall. A fall is detected if four phases namely standing, falling, fallen, and not moving are detected in sequence. Here, the untrimmed video was decomposed into sequences of video clips. These video clips were then converted to multiple dynamic images. Dynamic images are capable of storing both the appearance and temporal information of the clips. Some examples of dynamic images have been shown in Fig. 4.

These dynamic images were fed into a deep CNN to score and classify an event as falling or not falling. To calculate the temporal extent of the fall, the difference scoring method (DSM) was used. Three publicly available datasets namely, MCFD, HQFSD, Le2i FDD, and one dataset (YouTube fall dataset (YTFD)) created by the authors were used to evaluate the experiments. YTFD dataset contained 430 falling incidents and 176 normal activities. Data augmentation was also used to increase the dataset size. To augment the data, first, they flipped each dynamic image horizontally, then cropped the image from the four corners and center to the dimension 224×224. A pre-trained VGG-16 CNN [43] on ImageNet dataset [31] was utilized. The last fully-connected layer was replaced with a new fully-connected layer which contained four neurons. Each dataset was divided into four different non-overlapping parts. Two parts were used for training, one for testing and one for validation purposes. The initial training rate was 0.001 and it decreased to 0.1 after every 100 iterations and stopped at 300 iterations. The value of the momentum (Momt.) and weight decay (WD) were 0.9 and 0.0005 respectively. The sensitivity and specificity of the experiment used in four datasets are shown in Table 3.

The time to train the network was 1 h. Testing time was 0.5s for converting a video clip to a dynamic image, and 0.01s for obtaining scores.

Marcos et al. (2017) [44] and Hsieh et al. (2017) [45] utilized optical flow in their works. Marcos et al. used optical flow images [46,47] as input to the CNN network. These images only represent the motion of the consecutive frames and don’t consider other appearance-related information like brightness, color, contrast, etc. Hence, using optical flow images, a generic CNN model can be built to detect the fall. A generic model can work in different scenarios. Extracted features from the generic CNN were used as input to the fully connected neural network (FC-NN) which classifies it as “fall” or “no fall”. The steps used are shown in Fig. 5.

Here also a modified VGG-16 model was used. The input layer of VGG-16 was replaced to accept the stack of optical images as input. The size of the stack used here was 20. To compensate the inadequate availability of the datasets for fall detection, transfer learning [48,49] was used here. Similar to Fan et al. [41] a pretrained VGG-16 network on ImageNet [31] dataset was used to get a low level generic features. After that, the first layer of the network was modified to get optical flow as input. Modified network was retrained using UCF101 dataset [29]. In the final steps, CNN layers up to the first fully connected layer were frozen and the last two FC layers were fine-tuned using the dropout (DO) value of 0.8 and 0.9 to get a “fall” or “not fall”. URFD, MCFD, and the Le2i FDD were exploited. Each dataset was split into 80:20 ratios for training and testing respectively. Adam optimization (Optim.) was used

("Vision-based Fall detection") OR ("Fall accident") OR ("Video fall") OR ("motion fall") OR ("Fall")

(a) Part 1 Keywords

("Deep learning") OR ("LSTM") OR ("CNN") OR ("Neural Network") OR ("Transfer Learning") OR ("Auto-encoder") OR ("Autoencoder")

(b) Part 2 Keywords

(Part 1 Keywords) AND (Part 2 Keywords)

(c) Using AND between part 1 keywords and part 2 keywords

Fig. 2. Generation of papers for review.

Fig. 3. Basic steps in HFD.

Table 3
Result of the experiment by Fan et al. [41] using four different data set.

Dataset	Sensitivity	Specificity	Avg. Sensitivity	Avg. Specificity
Le2i FDD [21]	98.43	100	83.36	83.65
MCFD [22]	97.1	97.9		
HQFSD [34]	74.2	68.6		
YouTube fall dataset [41]	63.7	68.1		

Table 4
The result obtained by Marcos et al. [44].

Dataset	LR	MBS	wo	AF	Sensitivity	Specificity
URFD [23]	10^{-5}	64	1	ReLU	100%	94.86%
MCFD [22]	10^{-3}	Full	1	ReLU	98.07%	96.20%
FDD [21]	10^{-4}	1024	2	ELU	93.47%	97.23%

Table 5
Results of the experiment as reported by Zhang et al. [59].

Dataset	Sensitivity	Specificity	Accuracy	Training Time
SDUFall [24]	93.01	99.50	96.04	65 min
URFD [23]	100	95.00	–	30 Minutes
MCFD	90.21	92.59	–	35 Minutes
Average	**94.41**	**95.70**	**96.04**	43.33 Minutes

Table 6
Results of the experiment as reported by Espinosa et al. [8].

Camera	Model	Sensitivity	Specificity	Accuracy
Cam 1(Lateral view)	Proposed CNN	97.72	81.58	95.24
Cam 2(Front view)	Proposed CNN	97.57	79.67	94.78
Cam1 & Cam 2	Proposed CNN	97.95	83.08	95.64
Cam1 & Cam 2	VGG-16	100	0	84.44

in this work. This work was implemented using the Keras framework and is available at GitHub [1]. The learning rate (LR), mini batch size (MBS), activation function (AF), and other details are shown in Table 4.

Hsieh et al. [45] proposed optical flow feedback based CNN technique for fall detection. Here, the rule-based filters were used to filter out the inputs to the CNN. Falls were recognized by sequencing the frames of different actions.

Li et al. (2017) [20] proposed a fall detection system where CNN was applied to each frame of the video to extract human postures. As pre-processing of the data, the average image was computed over all training and test images, and then all the training, as well as test images, are subtracted by this average image to get a uniform brightness image. After this, contrast normalization for each image was done. The architecture used, is very similar to AlexNet [50]. The fully connected layer FC7 and output layer fc8 were replaced by a new fully-connected layer fc7 and an output layer fc8 with 1024 and 2 neurons respectively. "**MatConvNet**" [51], a toolbox written using Matlab, was used for the training purpose and **Gaussian distribution** was utilized for weight parameters initialization. Minibatch stochastic gradient descent was used with batch size 85. The initial LR used was 0.05 and was multiplied by 0.1 after 20 epochs (Epo.). 40 epochs were used for training. The publicly available dataset "URFD" was utilized. This dataset contains 30 fall sequences and 40 ADL sequences. The authors used all 30 fall sequences and only 28 ADL sequences out of 40.12 ADL sequences that are recorded in the dark environment were not used. An accuracy of 99.98 was reported.

Solback et al. (2017) [52] proposed a method to detect falls using depth images captured by Stereolab's ZED stereo cameras (left and right). 2D Human pose was estimated using CNN. MS COCO dataset was used to describe human body pose. 3D pose was generated using camera matrix, depth image, and key point information from 2D human pose estimation. Depth image was also used for 3D ground plane detection. Ground plane detection and 3D pose calculation were used for reasoning steps, which finally detects whether there is a fall or not. The true positive rate reported is 93.3%. This proposed system was implemented in Robot Operating System (ROS) [53] and available publicly [2].

Iuga et al. (2018) [54] proposed a fall detection system using unmanned aerial vehicle (UAV) [55,56] in an indoor setup. They used "Parrot AR.Drone 2.0" quadrotor [57] UAV which was built by a

[1] https://github.com/AdrianNunez/FallDetection-with-CNNs-and-Optical-Flow.

[2] https://github.com/TsotsosLab/fallen-person-detector.

Table 7
Fall Detection using CNN: Basic Details.

Reference	Sensor	Dataset	Technique	Framework
Doulamis et al. (2016) [39]	RGB	SCDS	TDNN	–
Fan et al. (2017) [41]	RGB	[21,22,34] [41],	Dynamic Images, VGG-16	Caffe
Marcos et al. [44]	RGB	URFD, MCFD, Le2i FDD	Optical flow	Keras
Li et al. [20]	RGB	URFD [23]	Postures, Avg Image	MatConvNet
Solbach et al. [52]	RGB	MS COCO [32]	Pose estimation, Depth Image	ROS
Hsieh et al. [45]	RGB	KTH [28]	Optical flow, rule based filter	–
Iuga et al. (2018) [54]	RGB	MS COCO, SCDS	YOLO v2, UAV	–
Zhang et al. [59]	RGB	SDUFall, URFD, MCFD	TDRD	C++, Python, MATLAB
Shen et al. [61]	RGB	Self created	Deep Cut NN	Keras
Kong et al. (2019) [63]	Depth	[63]	ETDA-Net	–
Cameiro et al. [65]	RGB	Imagenet, UCF101, URFD, Le2i FDD	Multi-Stream CNNs	–
Brieva et al. [67]	RGB	SCDS	Optical flow	–
Cai et al. [69]	RGB	URFD [23]	CC-MHI	–
Cai et al. [68]	RGB	URFD [23]	Optical flow	–
Kasturi et al. [71]	Depth	URFD [23]	3D CNN	Keras, Tensorflow
Espinosa et al. [8]	RGB	UP-fall detection [26]	Optical flow, 2D CNN	Keras 4, Sklearn 3
Leite et al. [66]	RGB (Video)	ImageNet, UCF101, URFD, Le2i FDD	Multi-Stream CNNs	–
Wu et al. [73]	RGB (Kinect)	SCDS, Activity3D dataset [30]	Skeleton sequence, light weight CNN	–
Zheng and Zhou [74]	RGB	NTU RGB + D, URFD	Pose Estimation, GCN	Pytorch
Espinosa et al. (2020) [76]	RGB	UP-fall detection [26]	optical flow	–
Chen et al. [80]	Depth	[63]	Edge Computing	–
Menacho et al. [79]	RGB	URFD [23]	optical flow	Keras
Carlier et al. [77]	RGB	MCFD, URFD, Le2i FDD	Optical flow, custom VGG-16	–
Yao et al. [78]	RGB	SCDS	Head Segmentation, Shallow CNN	–
Ijina [81]	Depth	SDUFall [24]	temporal template representation	–
Hader et al. [82]	RGB	Le2i FDD	Region based, faster R–CNN	Matlab
Dichwalkar et al. [84]	RGB	COCO, Robinovitch et al. [86]	SSD-MobileNet	Caffe
Lezzar et al. [88]	RGB	FPDS, SCDS	CNN and SVM	–
Zhang et al. [89]	RGB	SCDS (PoF)	M-CNN	–
Asif et al. [87]	RGB	COCO, SCDS, MCFD, Le2i FDD	SSHFD, OJR, SH	Torch Lib [97]
Asif et al. [91]	RGB	COCO, SCDS, MCFD, Le2i FDD	HPES, FallNet	Torch Lib [97]
Euprazia and Thyagharajan [36]	RGB	MCFD, URFD, Le2i FDD, SisFall	SCNN with transfer learning	Matlab2018a
Zhong et al. [90]	Thermal	Ulster University	Multi-occupancy	–
Liu et al. [92]	RGB	NTU-RGB, ISA	Two stream GCN	MMSkeleton [98]
Chen et al. [93]	RGB	Self created	OpenPose, Speed, Angle, Ratio	
Abdo et al. [95]	RGB	URFD [23], FDD [21]	RetinaNet, Handcrafted features, MobileNet	Keras, Tensorflow
Kareem et al. [94]	RGB	MCFD	OpenPose, ResNet-50	Google Colab
Chhetri et al. (2021) [99]	RGB	URFD [23], FDD, MCFD	Dynamic Optical Flow	Python
Chen et al. [100]	RGB	NTU RGB + D [33]	Pose estimation	–
Killian et al. [101]	RGB	SCDS	Cluttering awareness, YoloV3	NAOqi, Python, OpenCv
Leite et al. [102]	RGB	URFD [23], FDD [21]	Three stream 3D CNN	SciKit, OpenCV, Keras, TensorFlow
Zou et al. [103]	RGB	Le2i FDD, MCFD, LSST [103]	3D CNN	Caffe
Vishnu et al. [104]	–	Le2i FDD, URFD, Montreal	ResNet-101	–
Cai et al. [105]	RGB	URFD [23]	MCCF, dense block	–
Keskes and Noumier [106]	Skeleton data	NTU RGB-D, TST v2, Fallfree	ST-GCN, transfer learning	Pytorch
Berlin et al. [107]	RGB	URFD, Le2i FDD	Siamese CNN	Pyhton
Li et al. [108]	RGB	URFD, NTFD [108]	Saliency maps	Matlab 2014, Caffe

company named Parrot. YOLOv2 (You Only Look Once version 2) [58] object detection model was used to detect a human from the images captured by the AR. Drone 2.0. A pre-trained CNN with MS-COCO dataset was applied to detect a human in a standing position. Same YOLOv2 was used for fall detection with some fine-tuning in the last layer using a self created dataset (SCDS). The dataset contains 500 manually labeled images. These images were extracted from the frames of two videos. A single person wearing a fixed set of clothes was used as a subject for these recordings. For testing, 619 images were used with two subjects both wearing different clothes than the training images. All computation was done offline using a computer on the data received by the UAV wirelessly.

Zhang et al. (2018) [59] proposed a **TDRD** (Trajectory-weighted Deep-convolutional Rank-pooling Descriptor) based fall detection system. The process to get the TDRD is shown in Fig. 6. First, CNN feature maps were extracted from the input RGB video using slightly modified VGG-16 network [43]. Extracted feature maps were normalized using the spatio-temporal normalization method [60]. Then, improved dense trajectories were calculated from the input RGB video, which was used to get trajectories attention maps. Trajectories attention maps can locate the human regions. Next, for each frame, trajectory-weighted convolution features were calculated from the extracted normalized feature maps and trajectory attention maps. This trajectory-weighted convolution feature maps sequence of all frames can give information about human action dynamics. To reduce the redundancy, cluster pooling was applied on trajectory-weighted convolutional feature sequences. Finally, rank pooling was used on the cluster pooled sequence to get the TDRD.

The performance of the experiments and its average value using the three mentioned datasets are shown in Table 5.

Another fall detection system was presented by Shen et al. (2018) [61]. This system has three components namely (i) two cameras, (ii) a cloud server, and (iii) a smartphone. Cameras were used for taking the videos of the subject and converting it to high-frequency images. Raspberry Pi 3 Model B board was used for the cameras. Cloud server was used for processing the data and smartphone for receiving the alert about the falls. **Deepcut** neural network model [62] was used to identify the 14 key points of the human body. The detected key points data map was fed into the deep neural network for the detection of falls. Fig. 7 shows the structure of the depth neural network with human body key point distribution.

The authors used 44 pre-recorded videos, out of which 42 were used for training and 2 for testing. Two cameras with various angles, scenes, and poses were used. The average accuracy reported is 98.05%.

Kong et al. (2019) [63] showed how the height of the camera may

Table 8
Fall Detection using CNN: Evaluation Metrics.

References	Sensitiv.	Specific.	Accur.	GM	FS	RT	PvPr	Precis.	Notes
Fan et al. [41]	83.36*	83.65 *	83.86	–	–	Yes	–	–	*Average
Marcos et al. [44] (URFD)	100.0	92.00	95.00	–	–	–	–	–	–
Marcos et al. [44] (MCFD)	99.00	96.00	–	–	–	–	–	–	–
Marcos et al. [44] (FDD)	99.00	97.00	97.00	–	–	–	–	–	–
Li et al. [20]	100.0	99.98	99.98	99.99	0.0234	yes	No	–	–
Solbach et al. [52]	–	–	Above 91	–	–	–	–	–	–
Iuga et al. [54]	86.4	–	–	–	–	–	–	–	–
Zhang et al. [59]	94.41*	95.70*	96.04*	G-	–	–	No	–	*Average
Shen et al. [61]	–	–	98.05	–	–	–	–	–	–
Cameiro et al. [65] (FDD)	99.9	98.32	98.43	–	–	–	–	–	–
Cameiro et al. [65] (URFD)	100	98.61	98.77	–	–	–		–	–
Cameiro et al.(Average) [65]	99.95	98.46	98.6	–	–	–	–	–	–
Brieva et al. [67]	95.42	–	92.78	–	95.34	RT	PvPr	95.27	–
Brieva et al.(using majority voting) [67]	95.42	–	96.84	GM	–	RT	PvPr	95.27	–
Cai et al. [69]	–	–	92.34	GM	–	RT	PvPr	–	–
Cai et al. [68]	–	–	92.6	GM	–	RT	Yes	–	–
Kasturi et al. [71]	–	–	100*	GM	–	RT	PvPr	–	*maximum
Espinosa et al. [8]	97.95	83.08	–	–	–	–	–	–	–
Leite et al. [66](URFD)	100	98.77	98.84	GM	–	RT	PvPr	–	–
Leite et al. [66](FDD)	99.43	98.55	99.51	GM	–	RT	PvPr	–	–
Wu et al. [73]	93.9	–	93.75	GM	–	RT	PvPr	–	–
Zheng and Zhou [74]	97.1	–	94.1	GM	–	RT	PvPr	94.4	–
Espinosa et al. [76]	97.95	83.08	95.64	gm	–	rt	pp	96.91	–
Chen et al. [80]	92.884*	99*	97.352*	gm	–	rt	pp	–	*Average
Menacho et al. [79]	41.47	–	88.55	GM	–	Yes	PvPr	31.043	–
Carlier et al. (URFD) [77]	96.7	–	–	gm	–	rt	pp	85.3	–
Carlier et al. (FDD) [77]	90.9	–	–	gm	–	rt	pp	92.8	–
Carlier et al. (MCFD) [77]	71.0	–	–	gm	–	rt	pp	87.1	–
Carlier et al. (Avg) [77]	86.2	–	–	gm	–	rt	pp	88.4	–
Yao et al. [78]	–	–	90.5	GM	–	Yes	PvPr	–	–
Ijina [81]	–	–	91.58	GM	–		Yes	–	–
Hader et al. (VGG-16) [82]	99.4466	–	99.5	GM	–	–	PvPr	99.526	–
Dichwalkar et al. [84]	–	–	60	GM	–	–	PvPr	–	–
Lezzar et al. [88]	100	–		GM	–	–	PvPr	93.94	–
Zhang et al. [89]	–	–	98.7	GM	–	–	PvPr	–	–
Asif et al.(MCFD) [87]	84.31	–	–	GM	84.53*		PvPr	84.87	*F1 score
Asif et al.(Le2i) [87]	89.92	–	–	GM	89.91*		PvPr	90.08	*F1 score
Asif et al.(MCFD) [91]	98.61*	–	–	GM	98.60*		PvPr	98.60*	*Maximum
Asif et al.(Le2i) [91]	92.44*	–	–	GM	92.44*		PvPr	92.45*	*Maximum
Euprazia and Thyagharajan [36]	–	–	100	GM	–	–	PvPr	–	–
Zhong et al. [90]	–	–	95.89*	95.92 (±0.68)	–	–	PvPr	–	* (±0.50)
Liu et al. [92]	–	–	–	GM	–		PvPr	–	–
Chen et al. [93]	98.3	95	97	GM	–	RT	PvPr	–	–
Abdo et al. [95]	97.7*	100	98	–	–	–		–	*Approx
Kareem et al. [94]	–	–	97.46	GM	–	RT	PvPr	–	–
Chhetri et al. [99]	–	–	91.405*	GM	–	RT	PvPr	–	*Average
Chen et al. [100]	–	–	99.83	GM	–	RT	PvPr	–	–
Killian et al. [101]	–	–	88.88	GM	–	RT	PvPr	–	–
Leite et al. [102] (URFD)	0.99	–	0.99	GM	–	RT	PvPr	–	–
Leite et al. [102] (FDD)	0.99	–	0.99	GM	–	RT	PvPr	–	–
Zou et al. [103]	100.00	97.04	97.23	GM	–	RT	PvPr	–	–
Vishnu et al. [104] (Le2i)	93.0	–	–	GM	86.8	RT	PvPr	81.5	–
Vishnu et al. [104] (URFD)	76.6	–	–	GM	82.1	RT	PvPr	88.4	–
Vishnu et al. [104] (Montreal)	99.1	94.8	–	GM	–	RT	PvPr	–	–
Cai et al. [105]	95.0	100	96.6	GM	97.3	RT	PvPr	100	–
Keskes and Noumier [106] (TST v2)	–	–	100	GM	–	RT	PvPr	–	–
Keskes and Noumier [106] (Fallfree)	–	–	97.33	GM	–	RT	PvPr	–	–
Berlin et al. [107] (URFD)	–	–	100	GM	–	RT	PvPr	–	–
Berlin et al. [107] (FDD)	–	–	97	GM	–	RT	PvPr	–	–
Li et al. [108] (URFD)	–	–	99.67	GM	–	RT	PvPr	–	–
Li et al. [108] (NTFD)	–	–	98.92	GM	–	RT	PvPr	–	–

affect the performance of the fall detection system. They used an enhanced tracking and denoising Alex-Net (ETDA-Net) which is basically an AlexNet with some pre-processing added to improve the tracking performance and to reduce the noise in the image. They created a dataset using a depth camera for their work. Images were captured from five different heights. The images captured from different heights were used for training the model. At the time of testing, ETDA-Net detected the camera height and used the model which was trained on the same height for better performance. Kaid et al. (2019) [64] used CNN architecture to reduce the false positives cases of fall detection using the angle assistance algorithm. Here the task of CNN is to filter the fall alert as false, if the image for which the fall is detected contained a person sitting in a wheelchair. Here, the authors' goal is to detect whether the fall detected image contains a person sitting in a wheelchair. If the answer is yes, then no fall alert will be generated, and if no then a fall alert will be sent. The advantage of this study is that it reduces the false-positive cases. The disadvantage is that it's not a fall detection system, it only reduces the false positives.

Cameiro et al. (2019) [65] and Leite et al. (2019) [66] proposed a multi-stream fall detection method. Cameiro et al. [65] exploited

Table 9
Fall Detection using CNN: Optimization Details.

References	Optim.	MBS	LR	Momt.	WD	Epo.	Classi-fier	DO	AF
Fan et al. [41]	–	–	0.001	0.9	0.0005	300	Softmax	–	–
Marcos et al. [44]	Adam	64 to 1024	0.00001, 0.001, 0.0001	–	–	3000–6000	FC-NN	0.9, 0.8	ReLU, ELU
Li et al. [20]	SGD	85	0.05	0.9	0.0001	40	–	–	–
Iuga et al. [54]	SGD	4	0.0001	0.99	–	–	–	–	–
Shen et al. [61]	–	–	–	–	–	–	Softmax	–	tanh
Cameiro et al. [65]	Adam	powers of 2	10^{-3} to 10^{-5}	0.99	–	500	–	–	–
Brieva et al. [67]	SGD	16	0.01	0.9	–	500	Softmax		ReLU
Cai et al. [69]	SGD	–	0.0005	–	–	–	Softmax	0.5	–
Cai et al. [68]	–	16 to 256	10^{-3} to 10^{-5}	–	–	–	Softmax	–	ReLU
Kasturi et al. [71]	SGD	8	0.001	–	–	100	–	0.2	–
Espinosa et al. [8]	Adam	–	–	–	–	50	Softmax	–	ReLU
Leite et al. [66]	Adam	1024	0.0001	–	–	500	SVM	–	–
Wu et al. [73]	Adam	32	–	–	–	–	Softmax	–	Leaky ReLU
Menacho et al. [79]	RMS-prop	32	0.001	–	–	20	–	0.5	ReLU
Carlier et al. [77]	op	128, 256	0.001	–	–	–	–	–	ReLU
Yao et al. [78]	SGD	–	0.00001	–	–	500	Softmax	–	ReLU
Ijina [81]	–	–	–	–	–	–	SVM	–	–
Hader et al. (VGG-16) [82]	SGD	–	0.001	–	–	10	Softmax	–	ReLU
Lezzar et al. [88]	–	–	–	–	–	–	SVM	–	–
Asif et al. [87]	Adam	–	0.01	–	0.0005	300	–	0.5	ReLU
Asif et al. [91]	Adam	–	0.01	–	0.0005	150	Softmax	–	–
Euprazia and Thyagharajan [36]	SGD	45	0.001	0.9	–	30	Softmax	–	–
Zhong et al. [90]	Adam	32	0.001	–	–	–	Softmax	–	ReLU
Liu et al. [92]	SGD	64	0.01	0.9	0.0001	9	Softmax	–	–
Abdo et al. [95]	–	32	0.01	0.95	0.001	200	Softmax	–	ReLU
Kareem et al. [94]	RMS-Prop	–	–	–	–	10	Softmax	0.2	ReLu
Chhetri et al. (2021) [99]	–	–	–	–	–	–	Softmax	–	–
Chen et al. [100]	Adam	–	0.0001	–	–	20	–	0.25	ReLU
Leite et al. [102](FDD)	Adam	192	0.00001	–	–	500	SVM	0.5	–
Zou et al. (2021) [103]	SGD	8		0.9	0.00005	–	Softmax	–	tanh
Keskes and Noumier [106]	SGD	–	0.01	–	–	100	Softmax	0.5	–
Berlin et al. [107]	–	–	–	–	–	30	–	–	ReLU, Sigmoid
Li et al. [108]	–	–	–	–	–	–	–	–	ReLU, Sigmoid

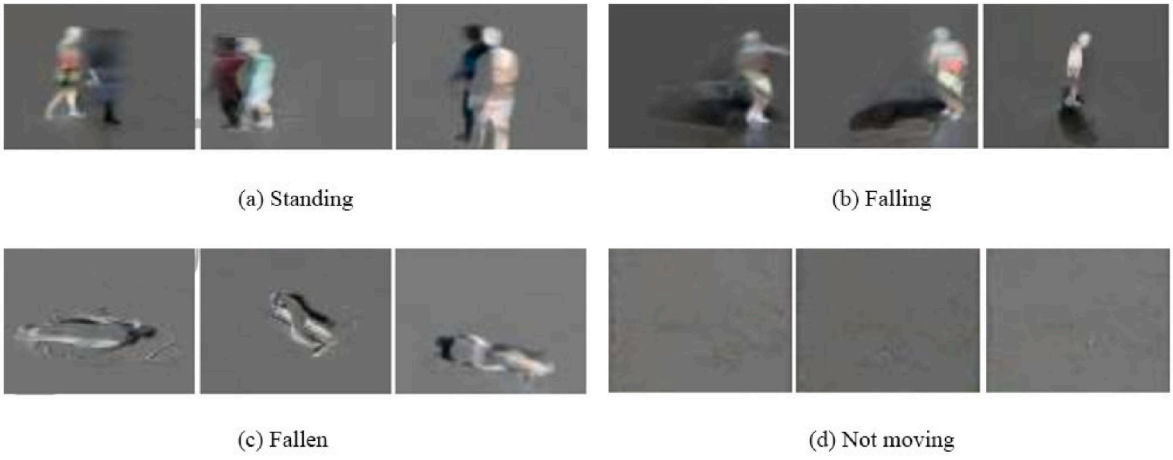


Fig. 4. Some examples of dynamic images of each phase used by Fan et al. from [41].

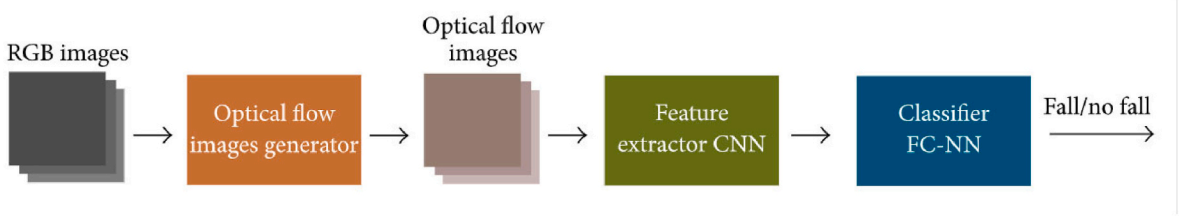


Fig. 5. System architecture used by Marcos et al. [44].

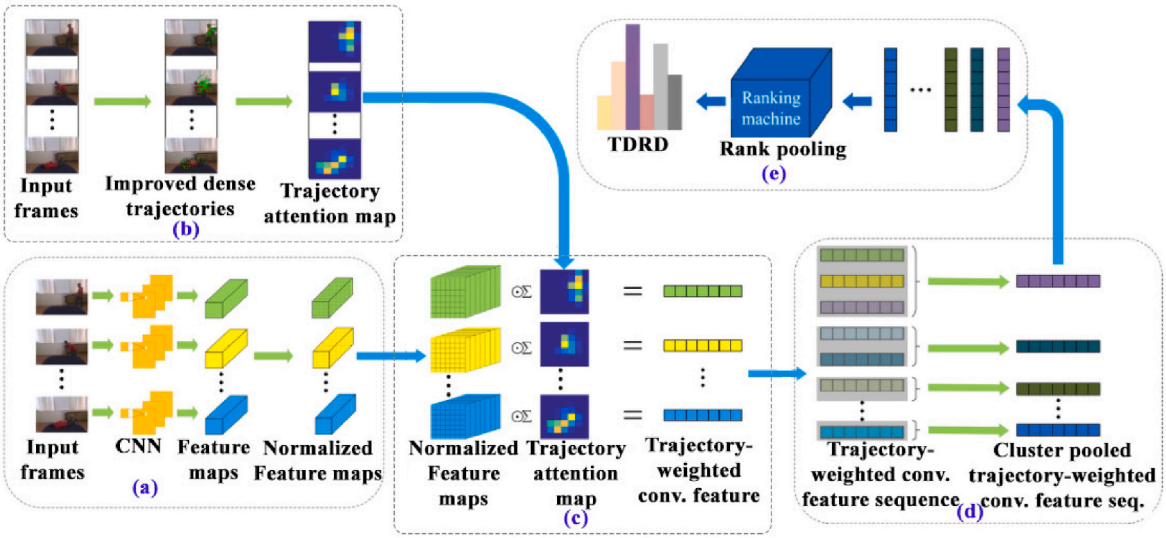
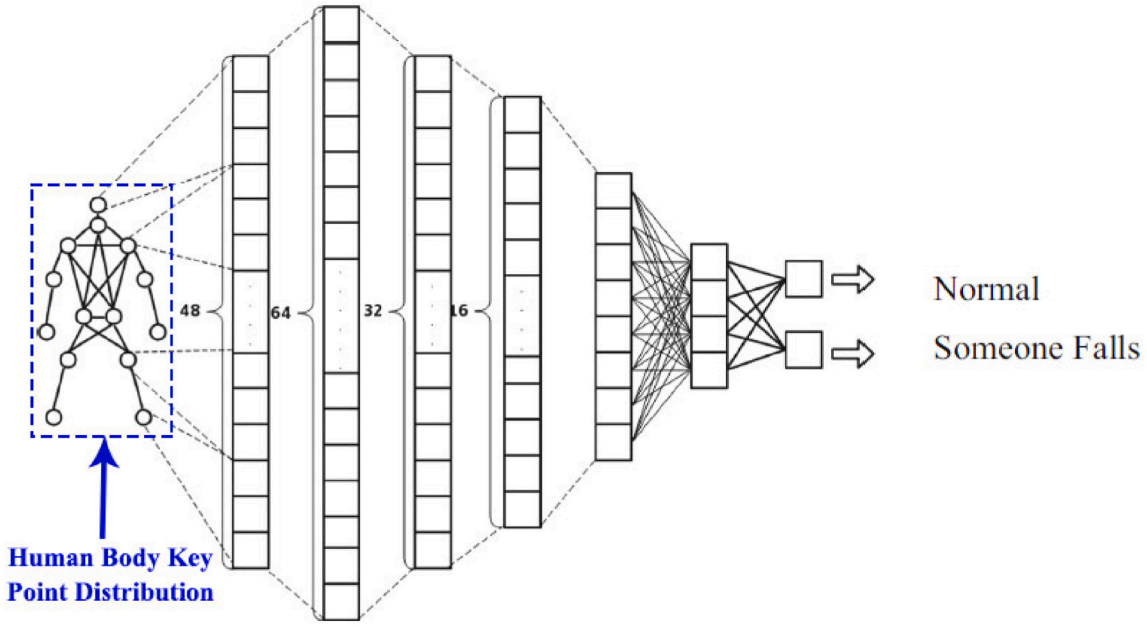


Fig. 6. The process of getting TDRD as proposed by Zhang et al.(2018). (a) Feature maps extraction and normalization. (b) Making trajectory attention maps. (c) Getting trajectory-weighted convolution feature from trajectory attention maps and normalized feature maps. (d) Cluster pooling on (c). (e) Finally getting TDRD from rank pooling on (d).



	Feature	Dense_1	Dense_2	Dense_3	Dense_4	Dense_5	Dense_6
Dimension	48	64	32	16	8	4	2
Activation		tanh	tanh	tanh	tanh	tanh	softmax

Fig. 7. Depth neural network structure as proposed by Shen et al. from [61].

handcrafted high-level features. The flow diagram of the proposed system is shown in Fig. 8.

Three independent CNNs (Modified VGG-16) with three different inputs (optical flow, RGB, and pose) from video sequences were used. Finally, the outputs of these three CNNs were ensembled to detect a fall. ImageNet dataset was used to train each individual CNN (VGG-16). The UCF101 dataset was used for optical flow generation and training the optical flow stream. URFD and Le2i FDD datasets were utilized for detecting a fall in video sequences. 80% of data were used for training and 20% for testing. The overview of the system proposed by Leite et al. [66] is shown in Fig. 9. They used optical flow, saliency map, and RGB as input for the three independent VGG-16 networks. Dual SVM (Support Vector Machine) was used for the classification and for weighting each stream. ImageNet and UCF101 were used for pre-training and fine-tuning respectively. For the experiments, URFD and Le2i FDD were utilized.

Similar to Marcos et al. [44], Hsieh et al. [45], and Cameiro et al. [65]; Brieva et al. (2019) [67], Cai et al. (2019) [68] and Espinosa et al. (2019) [8] also exploited optical flow for fall detection. Brieva et al. [67] created a fall detection dataset using a single subject. The length of each

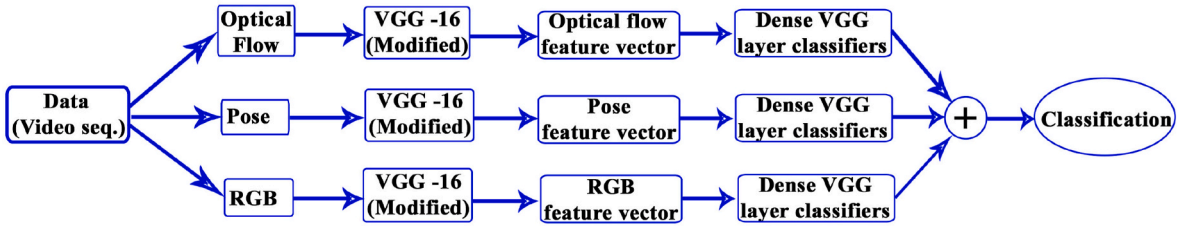


Fig. 8. Multi-stream fall detection as proposed by Cameiro et al.

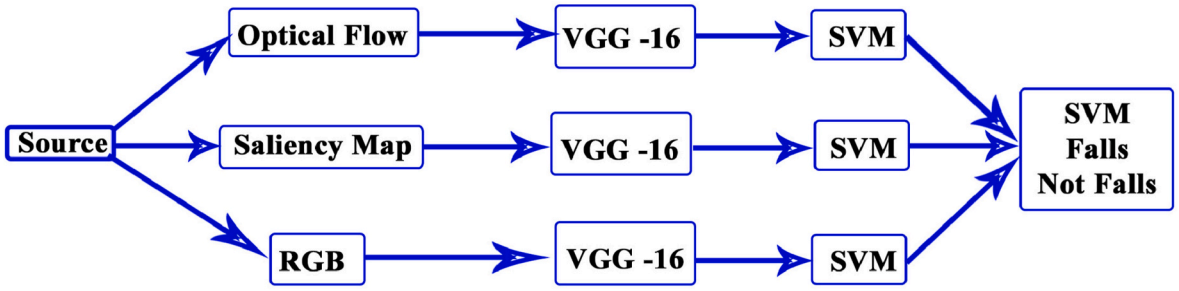


Fig. 9. The overview of the system as proposed by Leite et al. [66].

video recording was 10 s. The subject performed five types of falls namely (i) falling forward using hands, (ii) falling forward using knees, (iii) falling backwards, (iv) falling sitting in empty chair, (v) falling sideward. A CNN having a single layer with 25 filters, one Relu, one FC layer, one soft function, and one classification layer was used. The horizontal (I_u) and vertical components (I_v) of the calculated optical flow were combined as $I_{uv} = [I_u, I_v]$ which was used as input for the CNN. The images were resized to 28 × 56 size (pixels). The total 2817 images of the dataset were divided into two groups with 1972 (70%) images and 845 images (30%) for training and testing respectively. The value of regularization coefficient used was 1×10^{-4}. Cai et al. (2019) [68] used optical flow as input to the wide residual network. The overview of the proposed method by Cai et al. is shown in Fig. 10.

Espinosa et al. (2019) [8] exploited a 2D CNN model as shown in Fig. 11. Images from two cameras (lateral and front view) from the UP-fall dataset were used. Data was divided into 1-s time windows with 0.5-s overlap. Images were resized to 38 × 51 pixels grayscale images. 67% (42,000 images) of the data was used for training and 33% (21,000 images) for the testing. The optical flow was used as a pre-processing of the data. The experiments were done on a CNN architecture designed by the authors and on VGG-16.

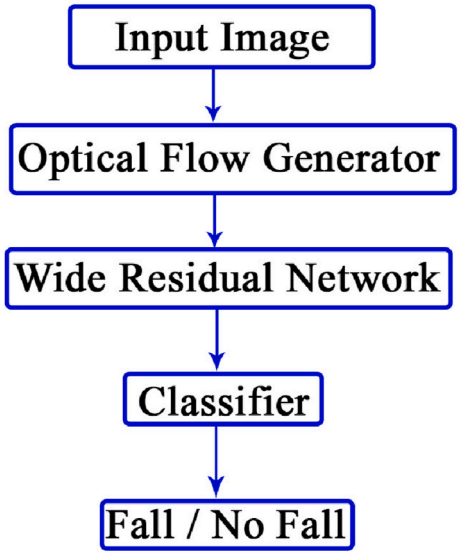


Fig. 10. The steps of the method as proposed by Cai et al. [68].

Experiments were executed in Python 3.7.3 using Keras 4 and Skleran 3 frameworks. In training, Adam optimizer was used with 50 epochs. The results of the experiments are shown in Table 6.

Cai et al. (2019) [69] proposed a fall detection system using colorization coded motion history image (CC - MHI) on VGG-16. The steps used in this method are shown in Fig. 12. Firstly, MHI [70] was calculated using the RGB image followed by color coding of MHI. This color-coded MHI was fed as input to VGG-16. Finally, the output was generated as fall or not fall. 1000 dimensional fully connected layer of VGG-16 was replaced with 2 dimensional fully connected layers.

Kasturi et al. (2019) [71] proposed a fall detection system using 3D-CNN [72]. Multi frame stacked image cube from Kinect camera was used as input to the 3D CNN. The overview of the system is shown in Fig. 13.

Dataset was divided into 80% and 20% for training and testing respectively. Wu et al. (2019) [73] proposed a human skeleton sequence-based fall detection system. The sequence of motion was encoded into an RGB image. They added their own dataset into the MSRDaily Activity3D dataset [30] to create a new dataset for their experiments. Kinect V2 was used for the data collection. All images were resized to 60 × 60. A lightweight CNN with 4 convolutional layers, 4 max-pooling layers, and 2 FC layers, was used to classify the encoded RGB image. LeakyReLU was used as an activation function. 1200 images of fall and non-fall and 300 images of fall and non-fall were used for training and testing respectively. Zheng and Zhou (2019) [74] proposed a fall detection system using pose estimation and GCN. The overview of the system is shown in Fig. 14. OpenPose [75] was used for pose estimation. Six categories (fall, jump up, sit down, stand up, stand down, and sit down) of data from the NTU RGB + D dataset [33] and URFD datasets were utilized.

A very similar work to Espinosa et al. (2019) [8] was proposed by Espinosa et al. (2020) [76] based on optical flow using two cameras. This work also used UP-Fall detection dataset. Carlier et al. (2020) [77] also proposed an optical flow-based fall detector for a Nursing Home environment using modified VGG-16. Yao (2020) et al. [78] proposed geometric features based fall detection system using a shallow CNN. At first, the head was segmented from the rest of the body and two different

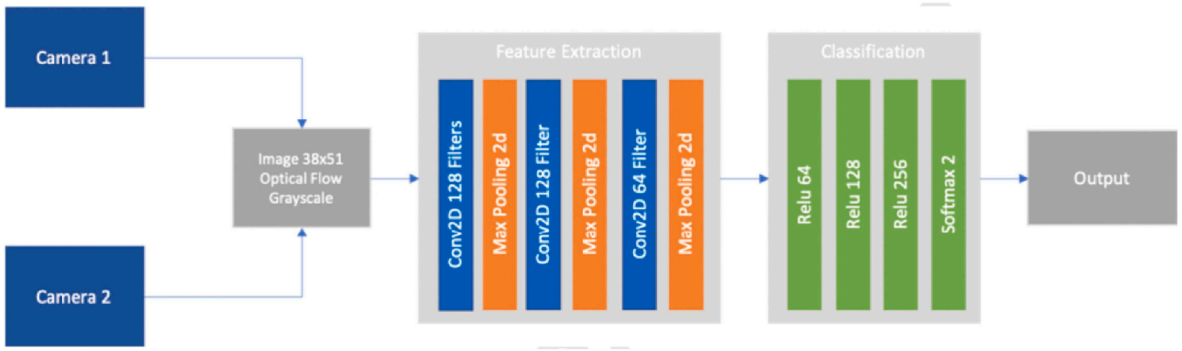


Fig. 11. CNN model as proposed by Espinosa et al. from [8].

Fig. 12. The overview of the system as proposed by Cai et al. [69].

Fig. 13. The overview of the system as proposed by Kasturi et al. [71].

Fig. 14. The overview of the system as proposed by Zheng and Zhou [74].

ellipses were generated for the head and torso respectively which were used to extract the long and short axis ratio, vertical velocity, and the orientation angle features. The authors used their own dataset which contains 102 videos, captured using multiple monocular cameras from different heights and angles. 74 videos were used for training and 28 videos for testing. Different height makes this system more robust. Menacho et al.(2020) [79] introduced an optical flow-based fall detection model using CNN with a reduced number of parameters that can be implemented in low computing devices like Mobile robots.

Chen et al. (2020) [80] proposed an edge computing-based fall detection technique and tried to analyze how the height and weight of the person can affect the performance. The basic parameters like height, weight, etc. were calculated using the edge node and sent to the cloud node for automatically selecting the best model. If any miss was detected, the missed image was also sent to the cloud for retraining the model. A subpart of the thermal dataset created by Kong el at [63]. was used. LeNet, AlexNet, and GoogleNet were exploited for the experiment. GoogleNet gave the best result. Ijina (2020) [81] proposed an HFD technique using temporal template representation of depth videos. Background subtraction was used as pre-processing. A pre-trained AlexNet was exploited. Hader (2020) et al. [82] proposed a technique for fall detection using region-based faster R–CNN [83]. Two pre-trained architecture, AlexNet and VGG-16, and three custom architecture were exploited. VGG-16 gave the best results.

Dichwalkar (2020) et al. [84] proposed a single shot detector (SSD) - MobileNet [85] model for fall detection. The frames of the video were resized to 300 × 300 and converted into blobs which were used as input to the SSD. For every detected object as a person, the difference of x coordinates and y coordinates were calculated. If the difference of x coordinates was greater than y coordinates then the object (person) either fell down or is in a sleeping position. The authors used the COCO dataset for training and the test videos from Ref. [86] for testing. Similar to Dichwalkar et al. [84], Asif et al. (2020) [87] proposed a single shot human fall detection (SSHFD) system which also considers some missing information in pose data using occluded joins resilience (OJR). At first, a bounding box of human was generated from a single RGB image which was fed as input to a stacked hourglass (SH) Network to generate a 2D pose. Using a neural network 2d pose was converted to a 3d pose. Finally, 2d pose and 3d pose were fed to another neural network for fall detection. MS COCO dataset was used for the training of the SH network. Authors created synthetic data was used for training the SSHFD, and for testing MCFD and the Le2i FDD were utilized. The initial LR used was 0.01 which was divided by 10 after 50% and 75% of the total number of epochs. Lezzar et al. (2020) [88] proposed a fall detection method considering more than one person in the frames and occlusions. To extract the persons (bounding boxes) YOLOv3 was used. The aspect ratio of the bounding box (AR), normalized bounding box width (NBW), and normalized bottom bounding box (NBB) were used as features for fall detection. CNN was exploited for features extraction and human detection; and SVM for recognition of posture. The FPDS, and a self-created datasets were used. Zhang et al. (2020) [89] proposed a fall detection model based on a “five-point inverted pendulum model” using a multi-stage convolutional neural network (M-CNN). Authors created their own dataset named “Postures of Fall (PoF)”. PoF contained single or multi-person RGB images captured using a single RGB camera. The overview of the proposed model by Zhang et al. is shown in Fig. 15.

Zhong et al. (2020) [90] proposed a multi-occupancy fall detection method using a thermal sensor. A pre-trained CNN was exploited. The

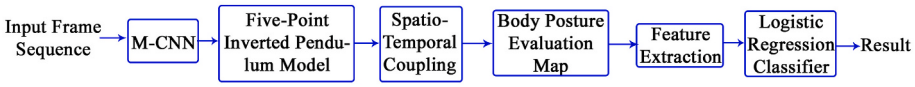


Fig. 15. The overview of the system as proposed by Zhang et al. [89].

images were divided into groups of 80% and 20% for the training and testing respectively. To collect the data a thermal sensor was installed on the ceiling of a room. Three persons (age: 25 to 35, one woman, two men, height: 1.68, 1.72, 1.83 m) performed different actions for data collection. Each captured thermal image was converted to 28 × 28 size. Both single and multi-occupancy data were recorded. For single occupancy, 186 data was labeled as fallen and 159 as non-fallen. For the multi-occupancy data, 528 images were labeled as fallen and 431 as non-fallen. To increase the dataset size data augmentation was used.

In another work, Asif et al. (2020) [91] proposed a human pose estimation and segmentation (HPES) based fall detection technique using multiple CNN structures. They used a single camera to capture RGB images. They used synthetic data to learn human proposals using body joint locations and segmentation information to detect falls. After training on the synthetic data they also used public MCFD and Le2i FDD to test their experiments. They represented an image to a skeleton-like structure. In this way, they removed all personal information like the face, etc of the subject and preserved the privacy. They also proposed a CNN model FallNet which uses the skeleton and segmentation-based representations and learns high-level features automatically for fall detection. The initial LR used was 0.01 which was divided by 10 after 50% and 75% of the total number of epochs.

Euprazia and Thyagharajan (2020) [36] proposed a method to recognize different human actions including falls. At first, action pattern image (API) was created from the input video which was fed to pre-trained series CNN (SCNN) to recognize the fall. To get the API at first edges of the frames were extracted using canny edge detection. The detected edges were combined into a single frame after proper enhancement. The LR used was 0.001 with a drop factor of 0.1 after every 8 epochs. Liu et al. (2020) [92] proposed a fall detection technique based on a 2D skeleton using two-streamed GCN. The overview of the proposed system is shown in Fig. 16. At first, 2D skeleton sequence was extracted using OpenPose. Some extra data was added by converting 3D skeleton to 2D skeleton from NTU-RGB + D dataset. Then, the skeleton sequence was converted to polar form. After that, two GCN were used in parallel one for polar and another one for Cartesian. Finally, the score of these two streams' GCN was combined to get the fall result. The authors created an indoor specific action (ISA) dataset.

Chen et al. (2020) [93] and Kareem et al. (2020) [94] presented a fall detection method using OpenPose. Chen et al. [93] used "speed of descent at the center of hip joint", "the human body centerline angle", and "width-to-height ratio" of the bounding box to detect the falls. Kareem et al. [94] utilized modified optimized ResNet-50 residual network.

Abdo et al. (2020) [95] proposed a fall detection method based on shape and motion features using RetinaNet [96] and MobileNet. RetinaNet was used to detect and track humans in video frames. Extracted handcrafted features like aspect ratio, head tracking, motion history images, head orientation were fed into MobileNet for final fall detection.

Chhetri et al. (2021) [99] proposed a fall detection system using enhanced dynamic optical flow. Three benchmark datasets namely URFD, MCFD, and Le2i FDD were used. At first, Input video was converted into RGB images. Then, the optical flow was calculated using those converted RGB images. After that, Rank pooling was calculated that was used to calculate the dynamic flow. Using the dynamic flow, the dynamic image was created. These steps were done as pre-processing. After pre-processing stage, feature extraction and classification were done. For this, a pre-trained VGG-16 (modified) was used. To make the fall detection more challenging, a variation of light was added to the input data. Results are shown in Table 10.

Chen et al. (2021) [100] introduced a pose-based fall detection technique. First of all, a 2D pose was extracted which was then converted to 3D pose. This 3D pose was fed to CNN for fall detection. NTU RGB + D dataset was exploited. The samples which have more than one person or some missing values were not used. Killian et al. (2021) [101] proposed fall prevention and detection technique using a bipedal humanoid NAO robot [109]. To prevent the fall a cluttering awareness technique was used. For this, the current image of the room of interest was compared with the standard reference image. If the cluttering level is high NAO robot speaks and asks to arrange the objects. To detect a fall YOLOv3 was used. If very little movement is detected in continuous images then there is a chance of fall. In this case, NAO robot asks some pre-defined questions to verify whether it's an actual fall or a false fall. If the person says that he/she needs helps or no response is received, then the robot sends the alerts to the caregivers using the Internet. The average accuracy in a lab environment was 94.44. The average highest and lowest accuracy in the real home environment was 88.88 and 83.33 respectively. The overall average accuracy was 88.88.

Table 10
Results of the experiment as reported by Chhetri et al. (2021) [99].

Dataset	Accuracy
URFD [23]	95.11
MCFD [22]	92.91
FDD [21]	91.1
FDD [21] (Dark lighting condition)	86.5
Average	**91.405**

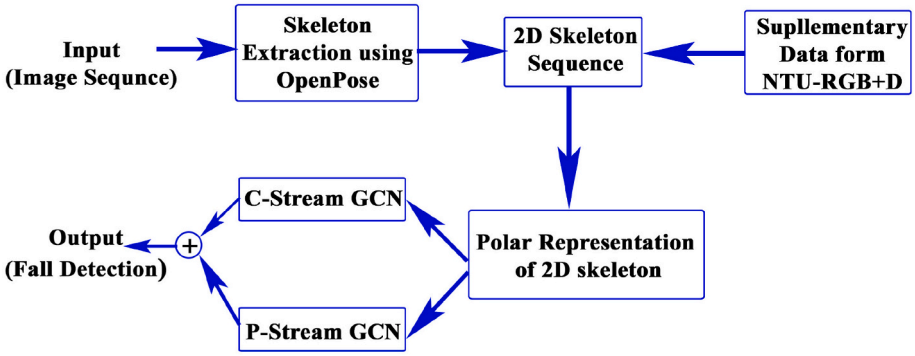


Fig. 16. The overview of the system as proposed by Liu et al. [92].

Leite et al. (2021) [102] introduced three-streamed 3D CNN network-based fall detection system. A pre-trained network on ImageNet was used for each stream. Three pre-trained models were trained again using Optical flow, Visual Rhythm, and pose estimation features independently. Finally, an SVM classifier was used for fall detection. URFD and Le2i FDD datasets were utilized with augmentation. Dataset was divided into 65%, 15%, and 20% for training, validation, and testing respectively. Zou et al. (2021) [103] presented fall detection method using 3D CNN. At first, the sequence of frames was fed as input to the 3D CNN to extract 3D features. A reshape layer converts the 3D features to the 2D features. Then, a spatial pyramid was used to generate multi-scale features which were fed to the movement tube regression layer, tube anchors generation layer, and softmax classification layer. After that, a matching and hard negative mining layer were used to find the tube anchors that match the ground truth using intersection-over-union (IOU). Finally, falls were detected using softmax and movement tube regression layer. SSD is used to detect objects. Apart from Le2i FDD [21] and MCFD [22], a large-scale spatial-temporal (LSST) dataset was created. Data augmentation was used by changing the spatial, illumination, and temporal properties. Vishnu et al. (2021) [104] proposed a fall motion vector based fall detection method using 3D CNN. A pre-trained ResNet-101 was used. Le2i [21], URFD [23] and Montreal [110] datasets were used. Berlin et al. (2021) [107] proposed a siamese neural network [111,112] based fall detection technique using single-shot classification [113]. Siamese network has two symmetrical convolutional networks. These two networks calculate the difference of input video sequence from another ADL video sequence and generate a distance value in the range of [0,1] where 1 denotes complete similarity and 0 denotes no similarity. Two filters, traditional 2D Conv. filter, and depth-wise Conv. filter were used. Two types of input features stacked RGB features, and optical flow features were utilized.

Cai et al. (2021) [105] introduced a multi-channel convolutional fusion (MCCF) based fall detection method using dense blocks and transition layers. At first, input frames (10) were fed to the convolutional layer. The output of the convolutional layer was given to the MCCF-Dense block. The result of the dense block was fed to the transition layer for down-sampling and reducing the data redundancy. After a few repetitions of dense blocks and transition layers, a final dense block was used. Finally, a fully connected layer was added. The output of the FC layer was fed to softmax for final fall detection. Li et al. (2021) [108] introduced fused saliency-based fall detection system. This system had two parts namely the saliency maps generation part and fall detection part. Two-stream CNN model one for global stream and another one for local stream were used for saliency maps generation. Generated fused saliency maps were fed as input to a five-layer CNN for final fall detection.

Keskes and Noumier (2021) [106] proposed a fall detection system using spatial-temporal graph convolutional networks (ST-GCN) [114]. NTU RGB-D [33], TST v2 [115], Fallfree [116] dataset were used with transfer learning technique. The ST-GCN network consisted of ST-GCN units where each unit contains a spatial GCN and temporal GCN. The ST-GCN was consists of 10 layers. The first four layers consisted of 64 output channels, the next three layers contained 128 output channels and the last three layers consisted of 256 output channels. The kernel size used was 9. The ST-GCN was pre-trained on NTU RGB-D dataset. After that, the first nine layers were frozen and retrained using TST v2 and fallfree datasets. The input data (Skeleton graph) was represented as tensor (N,3,T,25,M) where N is the batch size, 3 is the joint coordinates (x,y,z), T is the number of frames in one video clip, 25 is the total number of skeleton joints of each person and M is the number of person in the same clip. (N,3,T,25,M) was modified to (N*M, C,T,V) and was fed to spatial GCN to extract the spatial features. The extracted spatial features were fed to temporal GCN to extract the temporal feature vectors of the same joint. Finally, softmax was used on these feature vectors to detect falls.

5.2. LSTM or RNN (recurrent neural network) based techniques

In this section LSTM based fall detection techniques are reviewed. The details about the experiments discussed in this section are summarized in Tables 11–13. Hasan et al. (2019) [117] proposed a method to detect a fall from video data using RNN with 2 layers LSTM. Here, the 2D pose estimation using OpenPose [75] algorithm was done which gave body joints. The body joints (RKnee, LKnee, RHip, MidHip, LHip, RShoulder, LShoulder, and Neck) were selected and a 24 frame pose sequence timestep was used. After that, extracted pose vectors were fed into a 2-layer LSTM which finally detected a fall. The overview of the system as proposed by Hasan et al. is shown in Fig. 17.

For every 24 frame sequence an overlap of 16 frames (66.67%) was used. The architecture used is shown in Fig. 18.

For the 8 joints total 16 values were required. Batch normalization (BN) was used after input and after the last LSTM.

Jeong et al. (2019) [118] proposed an LSTM based fall detection technique for manufacturing industries. Similar to Ref. [117], here also OpenPose was utilized to get the skeleton data. Two datasets, URFD and SDUFall were exploited. The LSTM with 2 stacked layers and 256 hidden-layers features were used. The raw skeleton data was processed to get the speed of the human centerline coordinate (SHCLC). The raw data with SHCLC was used to get better accuracy results.

Another work for fall detection using attention-guided LSTM was proposed by Feng et al. (2020) [119]. Here the objective was to detect the falls in a complex background environment where more than one person can be in a frame. The authors created a complex scene fall dataset for their experiments. They also tested their model on the URFD dataset and MCFD dataset for comparison purposes. At first, they used YOLO v3 to detect pedestrians in the scene. Deep-Sort method [122] was used to track the detected object. Features were extracted from the tracked object using VGG-16 which were fed to attention-guided LSTM for detecting falls.

Romaissa et al. (2020) [120] proposed a LSTM based fall detection using human body geometry. Down-sampling of the video clip was done using optical flow. The angle and the distance between the vector from the centroid of the head and the center of the hip with respect to the horizontal axis were calculated. These values were used to train an SVM and LSTM for fall classification. The Le2i FDD fall dataset was exploited. Two approaches were used: (i) Using angles, distance, Resnet50, SVM, and (ii) Using angles, distance, LSTM. Taufeeque et al. (2021) [121] introduced a multi-person and multi-camera-based fall detection technique. They used the UP-Fall dataset which contains data captured by two cameras. Since UP-Fall does not have multi-person data, they also created a dataset using two cameras that contain multi-person. First of all, human pose estimation was done using OpenPifPaf [123]. Then, multi-person was tracked by mapping keypoints using Gale-Shapley algorithm [124]. After that, geometrical (angle, aspect ratio, etc.) features were extracted. Then, LSTM was used to detect falls from the extracted features. If a fall was detected then some post-processing was done to reduce false positives.

Table 11
Fall Detection using RNN: Basic Details.

Reference	Sensor	Dataset	Technique	Framework
Hasan et al. (2019) [117]	RGB	MCFD, URFD, Le2i FDD	Pose estimation with LSTM	–
Jeong et al. [118]	RGB	URFD, SDUFall	Skeleton extraction, SHCLC	TensorFlow
Feng et al. (2020) [119]	RGB	SCDS, URFD, MCFD	Attention guided LSTM, YOLO V3, VGG-16	–
Romaissa et al. [120]	RGB	Le2i FDD	Body Geometry, Optical flow	–
Taufeeque et al. (2021) [121]	–	UP-Fall, SCDS	Pose estimation	PyTorch

Table 12
Fall Detection using RNN: Evaluation metrics.

References	Sensitivity	Specificity	Accuracy	GM	FS	RT	PvPr	Precision
Hasan et al. (MCFD) [117]	98.0	96.0	–	GM	–	RT	PvPr	–
Hasan et al. (Le2i FDD) [117]	99.0	97.0	–	GM	–	RT	PvPr	–
Hasan et al. (URFD) [117]	99.0	96.0	–	GM	–	RT	PvPr	–
Hasan et al. (Average)	98.67	96.33	–	GM	–	RT	PvPr	–
Jeong et al. [118]	–	–	98.83	GM	–	RT	PvPr	–
Feng et al. [119] (SCDS)	83.5	–	–	gm	86.5	rt	pvpr	89.8
Feng et al. [119] (URFD)	91.4	–	–	GM	93.1*	rt	PP	94.8
Feng et al. [119] (MCFD)	91.6	93.5	–	–	–	–	–	–
Romaissa et al. (i) [120]	90	–	84.60	GM	0.90*	RT	PvPr	90.00
Romaissa et al. (ii) [120]	89.00	–	84.60	GM	0.89*	RT	PvPr	89.00
Taufeeque et al. [121]	95.62	–	98.22	GM	92.56	RT	PvPr	89.76

Table 13
Fall Detection using RNN: Optimization Details.

References	Optimization	MBS	LR	Momentum	Weight Decay	Epochs	Classifier
Hasan et al. [117]	Adam	256	5×10^{-4}	–	–	350	Softmax
Jeong et al. [118]	Adam	–	0.0001	–	–	500	Softmax
Feng et al. [119]	Adam	15	0.0006	–	–	–	Softmax
Taufeeque et al. [121]	–	–	–	–	–	–	Softmax

5.3. Auto-encoder

In this section fall detection using auto-encoders is reviewed. The details about the experiments discussed in this section are summarized in Tables 14–16. In the year 2018, Doulamis and Doulamis [125] proposed a self-adaptable fall detection system using a sparse auto-encoder. This work considered abrupt visual changes like shadows, illumination, background, etc. This model adapts itself to the changing conditions while preserving the current knowledge. At first, humans were detected from the visuals. To do so, sparse auto-encoders with some supervised classification methods were used for the training of the network. To make the network adaptable, approximate methods were used to automatically select the most useful data of the current environment. To conduct the experiment, two datasets, one form mitseva et al. [126] and one created by the authors themselves, were utilized. The accuracy reported on these two datasets was 85% and 93% respectively.

Nogas et al. (2018) [127] introduced a thermal image based fall detection method using convolutional LSTM auto-encoder. TSFD dataset was used with data augmentation. The dataset was normalized by dividing each pixel value by 255 to get the value within the range of [0, 1] and subsequently subtracting the resultant pixel value by the mean of the pixels values per frame to get it in the range of [−1, 1]. Elshwemy et al. (2020) [128] proposed a Spatio-temporal residual auto-encoder (SRAE) model for fall detection using TSFD dataset. A ConvLSTM was utilized for this experiment. Nogas et al. (2020) [129] proposed a similar method related to Ref. [127]. Fall detection was considered as anomaly detection from ADL. Deep spatio-temporal convolutional auto-encoder (DSTCAE) was used to extract spatial and temporal features from ADL. Frames were encoded by the DSTCAE using 3D Conv and 3D Max pooling. Decoding was done using 3D UpSampling and 3D Convolution. Thermal and depth cameras were used which can protect the privacy of the subject.

5.4. MLP based techniques

A significant work for fall detection using MLP was proposed by Safarzadeh et al. (2019) [130]. They introduced a two-stage real-time fall detection system. In stage 1, pose estimation was done using a method proposed by Wei et al. [131]. After pose estimation, an MLP was used to classify these poses as falls or not falls. If a fall is detected an SMS is sent to the caretakers for the necessary action. The prototype system was built using the Arduino Uno board. To detect the joint even in a lying position, for each image extra two images have been generated by rotating it by an angle of −90 and 90°. Two fully connected layer MLP with Relu activation functions was used. The authors created their own dataset for their experiments. The dataset contains 250 images in lying positions in different poses and 250 images in standing, squatting, sitting, and other non-lying positions. Adam optimizer was used for the training. The network was trained till 100 epochs. The maximum accuracy reported is 94%. Ramirez et al. (2021) [132] introduced a fall detection method using human skeleton extraction. The details about the experiments discussed in this section are summarized in Tables 17 and 18.

5.5. Hybrid model based techniques

In many works, more than one type of DL model was used. Those works have been reviewed in this section as hybrid moles. The details about the experiments discussed in this section are summarized in Tables 19, 21 and 22.

Feng et al. (2014) [38] in their work used a single USB camera to capture the images. The background of the image was removed using a codebook background subtraction algorithm to extract the human body image. Some post-processing was applied in this extracted foreground image to remove noises due to shadows etc. Extracted binary silhouettes were fed into deep learning classifiers. Then, the silhouettes were compared with four postures namely standing, sitting, bending, and lying, to detect the falls. For classification, both deep belief network (DBN) and restricted boltzmann machine (RBM) were used. A detection rate of 86% and a false detection rate of 3.7% were reported.

Lu et al. (2017) [133] developed a three dimensional convolutional neural network (3D CNN) based fall detection system. Kinetic data were used for the training of the automatic feature extractor. To focus on the key regions, an LSTM based visual attention method was used. The input videos were split into clips. Each clip was of 16 frames with some overlapping frames. The experiment was done with 1, 4, and 8 overlapping frames. The results of the experiment are shown in Table 20. 3D CNN and LSTM were trained separately. **Sport-1M** dataset was used to pre-trained the 3D CNN. The 3D feature cube generated through the pre-trained 3D CNN was used as the input to train the LSTM based attention model.

A similar work to Ref. [133] was done by Lu et al. (2018) [145]. Abobakr et al. (2018) [134] proposed a fall detection system using **depth images** captured by MS Kinect RGB-D sensor [146]. This system

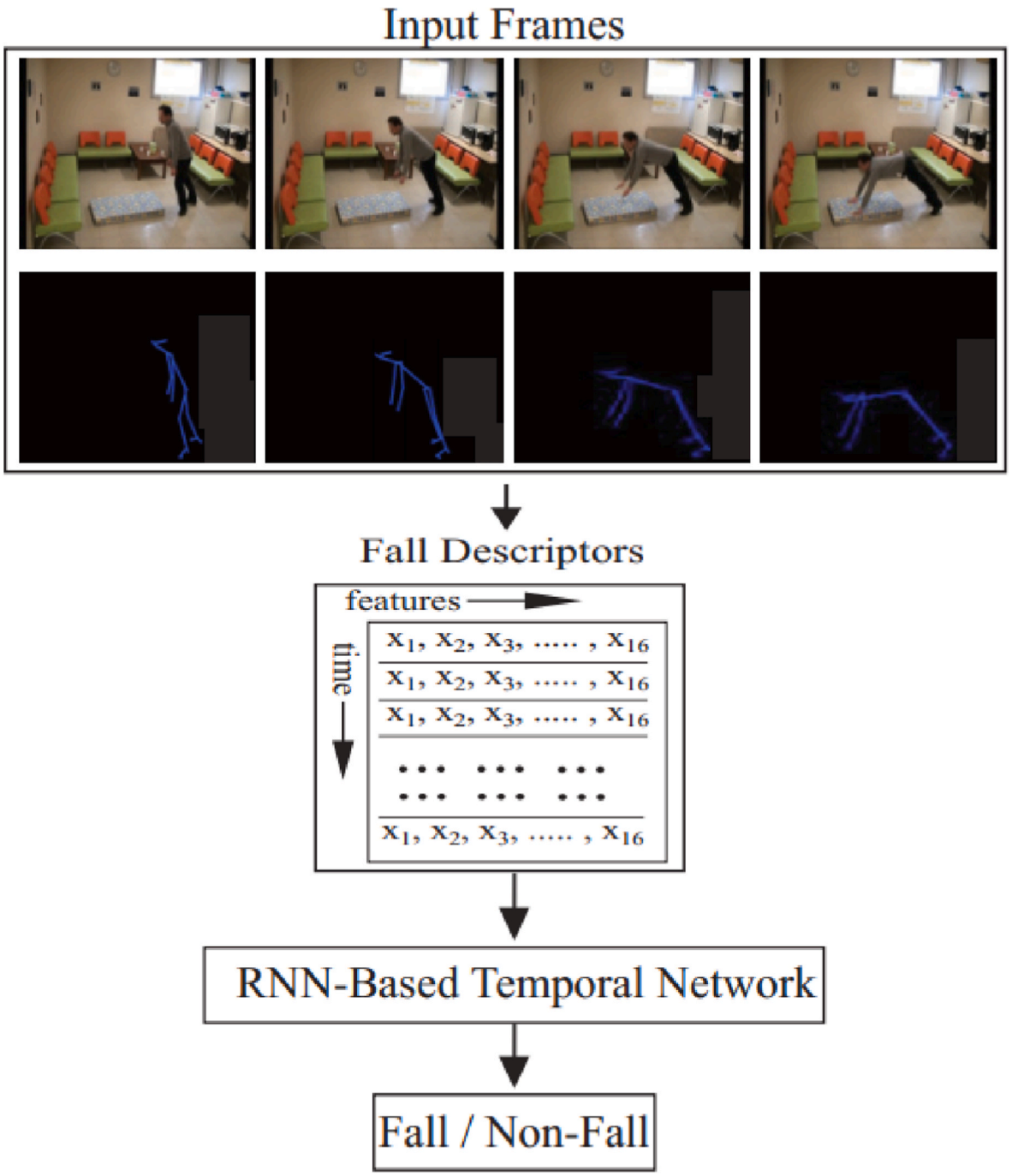


Fig. 17. The overview of the system as proposed by Hasan et al. from [117].

used deep CNN and RNN to detect falls. Deep CNN was used to extract the visual features from the input frame sequences. Authors followed the residual learning approach (ResNet) [147] for the CNN. On the top of the ResNet LSTM [148] recurrent neural network was used. LSTM gave the temporal dynamics information which discriminates an event as "fall" or "no fall". URFD was exploited for the training and evaluation of this system. 0.0005 was used as initial LR with a decaying factor of 10 after every 30 epochs. The overview of the proposed fall detection system is shown in Fig. 19.

Ma et al. (2019) [135] proposed a privacy-preserving fall detection system using 3D CNN (C3D) and auto-encoder (AE). For preserving privacy they used an optical level anonymous image sensing system (OLAISS) [149]. The advantage of using OLAISS is that it hides (black-outs) the facial regions at the time of capturing instead of doing this in post-processing. C3D was pre-trained using the Sports-1M dataset. To evaluate their performance they created their own dataset with two cameras one for side view and another one for front view recording simultaneously. Zhou and Komuro (2019) [136] proposed a fall detection system using a variational auto-encoder (VAE) [150] with 3D CNN residual blocks [151]. Here, the region with a human were extracted (cropped) and aligned at the left shoulder. They used unsupervised techniques with some weakly labeled data. Reconstruction error (mean square error) was used to detect falls. If the error was within the normal range, it was detected as normal ADL. If it was outside of the range, it was a fall. AlphaPose [152] was used to extract the region of human motion. The HQFSD and the Le2i FDD were used. 8 residual blocks of ResVAE-18 or 16 residual blocks of ResVAE-34 were used in the encoder and decoder of the VAE. For training the ResVAE-18, and ResVAE-34,

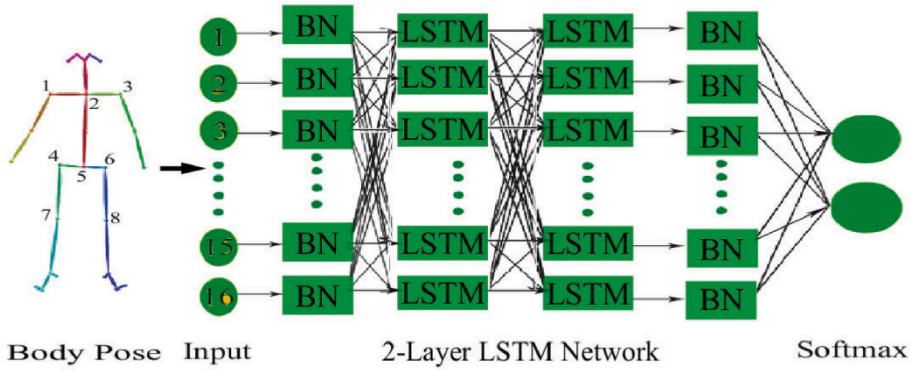


Fig. 18. The overview of the architecture as proposed by Hasan et al. from [117].

Table 14
Fall Detection using Auto-encoder: Basic Details.

Reference	Sensor	Dataset	Technique	Framework
Doulamis et al. (2018) [125]	–	ISESMD [126], SCDS	Sparse auto-encoder	OpenCV
Nogas et al. (2018) [127]	Thermal	TSFD [35]	Conv. LSTM AE	–
Elshwemy et al. (2020) [128]	Thermal	TSFD [35]	SRAE, ConvLSTM	Keras, Tensorflow
Nogas et al. (2020) [129]	Depth	TSFD, SDU, URFD	DeepFall, DSTCAE	–

Table 15
Fall Detection using Auto-encoder: Evaluation metrics.

Reference	Sensitivity	Specificity	Accuracy	GM	F Score	RT	ROC	Precision
Doulamis et al. [125] (ISESMD)	–	–	85	–	–	–	–	–
Doulamis et al. [125] (SCDS)	–	–	93	–	–	–	–	–
Doulamis et al. [125] (Average)	–	–	89	–		–	–	–
Nogas et al. [127]	–	–	–	–	–	–	0.83	–
Elshwemy et al. [128]	–	–	–	gm	–	r-	0.97	–

Table 16
Fall Detection using Auto-encoder: Optimization Details.

Reference	Optim.	MBS	LR	Momen-tum	Weight Decay	Epo.	Classifier	DO	Activ. Func.
Nogas et al. [127]	Adadelta	16 (8 frames per BS)	LR	Momen	Weight D	50	clsfr	–	–
Elshwemy et al. [128]	Adadelta	32	lr	mt	wd	30	cls	–	–
Nogas et al. [129]	Adadelta	16 (8 frames per BS)	LR	Momen-tum	Weight D	500	clsfr	0.25	ReLU (Encod.), tanh (Decod.)

the 12266 samples (ADL) of the HQFSD dataset were exploited. The testing of the ResVAE-18 and ResVAE-34 was done using 282 fall samples and 300 ADL samples from the HQFSD dataset and 130 fall samples and 200 ADL samples from the Le2i dataset.

Cai et al. (2020) [137] proposed a multi-task hourglass convolutional auto-encoder (HCAE) based fall detection system. Multi-scale features were extracted using an hourglass residual unit (HRU). Manekar et al. (2020) [138] proposed a fall detection method based on 3D CNN and LSTM. They used two techniques, one using 3D CNN and another one 3D CNN combined with LSTM. They created a 360-degree dataset using an omnidirectional camera. Chen et al. (2020) [139] proposed an attention guided bi-directional LSTM based fall detection method in complex scenes environment. URFD and a self-created dataset were exploited. To make the background complex Gaussian noise was added into the authors' created dataset. For background subtraction, Mask R–CNN [153] was used. After removing the background, features were extracted using the output of the last Conv layer of VGG-16. Finally, Extracted features were fed into attention-guided Bi-directional LSTM for fall detection.

Pourazad et al. (2020) [140] introduced a fall detection system using CNN and LSTM. A self-created RGB video dataset was used. A pre-trained Inception V3 (modified) network on the ImageNet dataset was used to extract features. The last output layer of the inception model was replaced by a 2048 unit FC layer. The extracted features were fed to three-layer LSTM for final fall detection. Kinects camera was used for video capturing. Li et al. (2020) [141] classified a fall detection technique into three types [141]. Frame-level fall detection (FLFD), Sequence-level fall detection (SLFD) and video-level fall detection (VLFD). Li et al. [141] presented a multi-level dynamic pose motion (DPM) based fall detection method using CNN-LSTM. They labeled the five public datasets frame-wise.

Mehta et al. (2021) [142] proposed a thermal-image-based fall detection system using two-channel 3D auto-encoders and 3D CNN. First of all, region of interest (ROI) extraction and optical flow computation were done from the thermal input frames. Thermal frames and optical flow frames with ROI masking were fed to thermal auto-encoder and

Table 17
Fall Detection using MLP: Basic Details.

References	Sensor	Datasets	Techniques	Framework
Safarzadeh et al. (2019) [130]	RGB	SCDS	Pose estimation,	–
Ramirez et al. (2021) [132]	RGB	UP-Fall	Skeleton extraction	–

Table 18
Fall Detection using MLP: Evaluation metrics.

References	Sensitivity	Specificity	Accuracy	GM	F Score	RT	PvPr	Precision
Safarzadeh et al. [130]	–	–	94	GM	–	Yes	PvPr	–
Ramirez et al. [132]	94.57 ± 1.15	98.21 ± 0.29	97.39 ± 0.10	GM	94.21 ± 0.27	RT	PvPr	93.87 ± 0.85

Table 19
Fall Detection using Hybrid Models: Basic Details.

Reference	Sensor	Dataset	Technique	Framework
Feng et al. (2014) [38]	RGB	SCDS	DBN, RBM	–
Lu et al. (2017) [133]	–	Sport1-M, MCFD	3D CNN, LSTM, spatial visual attention	–
Abobakr et al. (2018) [134]	Kinect RGB-D	URFD [23]	CNN, LSTM	–
Ma et al. (2019) [135]	RGB, Thermal	Sports-1M, SCDS	3D-CNN and auto-encoder	–
Zhou and Komuro [136]	RGB	HQFSD, Le2i FDD	VAE with 3D CNN, Residual blocks	–
Cai et al. (2020) [137]	RGB	URFD [23]	HCAE	Tensorflow
Manekar et al. [138]	RGB	SCDS	3D CNN and LSTM	–
Chen et al. [139]	RGB	URFD [23], SCDS	R–CNN, bi-directional LSTM	–
Pourazad et al. [140]	RGB	Self Created	Inception v3, LSTM	Keras
Li et al. [141]	RGB	[22, 21, 23, 34]	Multi-level, DPM	–
Mehta et al. (2021) [142]	Thermal	TSFD [35]	Auto-encoder, 3D CNN	–
Lin et al. [143]	–	URFD, Le2i FDD	RNN, LSTM, GRU, OpenPose	Cafe, Keras 2.3
Apicella and Snidaro [144]	RGB	URFD, MCFD	Pose estimation, CNN, LSTM	–

Table 20
Results of the experiment reported by Lu et al. [133].

No of overlapping frames	Sensitivity	Specificity	Accuracy
1	96.65	99.85	99.73
4	86.21	99.56	99.07
8	65.57	98.37	97.27
Average	**82.81**	**99.26**	**98.69**

flow auto-encoder respectively. The outputs of these two auto-encoders were given to the 3D-CNN thermal and flow discriminator. Finally, the fall was detected using the Joint discriminator of thermal and flow discriminator. Region-based fully convolutional network (R–FCN) [154], Otsu thresholding [155], Kalman filtering were used for person detection, Contour box localization, and tracking respectively.

Lin et al. (2021) [143] presented OpenPose based fall detection method using RNN, LSTM and gated recurrent unit (GRU). At first, sequences of continuous images were extracted from the input dataset. Then, the skeletons were generated from these sequences of images. After that, some pre-processing was done. Then, these pre-processed data were trained and tested. The URFD and Le2i FDD datasets were exploited after mixing these two datasets. If some part was missing due to overlapped or obscured, the linear interpolation was tried. The performance of linear interpolation was not so good. The results are shown in Table 23.

Romaissa et al. (2021) [156] proposed human body geometry-based fall detection method using LSTM and SVM classifier. Two geometric features were used, one was the angle between the line from the center of the head to the center of hip and horizontal axis, and another feature was the distance from the vector forming from the center of the head to the center of the heap and the vector forming from the horizontal axis. URFD and Le2i FDD dataset were exploited. At first, the down-sampling of the input video was done by calculating its optical flow. After that, manual annotations of the body and head were done. Then, angle and distance were calculated. Finally, falls were detected using a bidirectional LSTM and an SVM. Different combinations of architectures (Resnet50, AlexNet) and features (angle only, angle, and distance) were used. The performance using feature images + ResNet50 + SVM is shown in Table 24.

Apicella and Snidaro (2021) [144] presented a fall detection method using LSTM and CNN. At first, pose estimation was done using PoseNet [157]. If pose estimation was done accurately then extracted series of poses (20 poses) were fed to LSTM. If pose was not generated successfully or some keypoints (among 17 keypoints) were missing then an additional CNN was used to generate the series of poses. Finally, LSTM classified an activity as fall or no fall.

6. Discussions on limitations and future scope

The maximum work for fall detection using DL have been done using

Table 21
Fall Detection using Hybrid models: Evaluation Metrics.

References	Sensit.	Specif.	Accur.	GM	F S	RT	PvPr	Precis.
Feng et al. [38]	–	–	86	15		No	–	–
Lu et al. [133]	82.81	99.26	98.69	RT	–	PvPr	–	–
Abobakr et al. [134]	100	97	98	GM	–	Yes	Yes	–
Ma et al. [135]	0.933	0.928	–	GM	–	RT	Yes	–
Zhou and Komuro (ResVAE-18-HQFSD) [136]	–	–	88.7	GM	–	RT	PvPr	–
Zhou and Komuro (ResVAE-18-Le2i) [136]	–	–	87.3	GM	–	RT	PvPr	–
Zhou and Komuro (ResVAE-34-HQFSD) [136]	–	–	82	GM	–	RT	PvPr	–
Zhou and Komuro (ResVAE-34-Le2i) [136]	–	–	87.6	GM	–	RT	PvPr	–
Cai et al. [137]	100	93	96.2	GM	96	rt	pp	92.3
Manekar et al. (3D CNN) [138]	–	–	95.3		–			–
Manekar et al. (CNN-LSTM) [138]	–	–	91.63		–			–
Chen et al. (URFD) [139]	91.8	100	96.7	GM	94.8	RT	PvPr	100
Chen et al. (Self dataset) [139]	92.3	–	–	GM	94.8	RT	PvPr	98.1
Pourazad et al. [140]	–	–	87	GM	–	RT	PvPr	–
Apicella and Snidaro [144]	80.1	SP	81.8	GM	80.5	RT	PvPr	80.9

Table 22
Fall Detection using Hybrid Models: Optimization Details.

Reference	Optim.	MBS	LR	Momt.	WD	Epo.	Classifier	DO	AF
Lu et al. [133]	Adam	–	–	–	–	–	–	–	–
Abobakr et al. [134]	SGD	160 frames	0.0005	0.9	0.0001	100	Softmax	–	–
Ma et al. [135]	AdaGrad	28	10-^4	–	–	10	Softmax and SVM	–	–
Zhou and Komuro (ResVAE-18) [136]	Adam	32	–	–	–	500	–	–	–
Zhou and Komuro (ResVAE-34) [136]	Adam	24	–	–	–	500	–	–	–
Manekar et al. (3D CNN) [138]	Adam	128	.0001		–	300	Softmax	0.25	–
Manekar et al. (CNN-LSTM) [138]	Adam		0.00001	–	–	60	–	–	–
Mehta et al. [142]	SGD, Adadelta	–	0.0002	–	–	300	–	–	–
Lin et al. [143]	Adam	916	0.1, 0.01, 0.001	–	–	500	–	–	Tanh
Apicella and Snidaro [144]	Adam	–	0.0001	–	–	1000	Sigmoid	0.2	ReLU

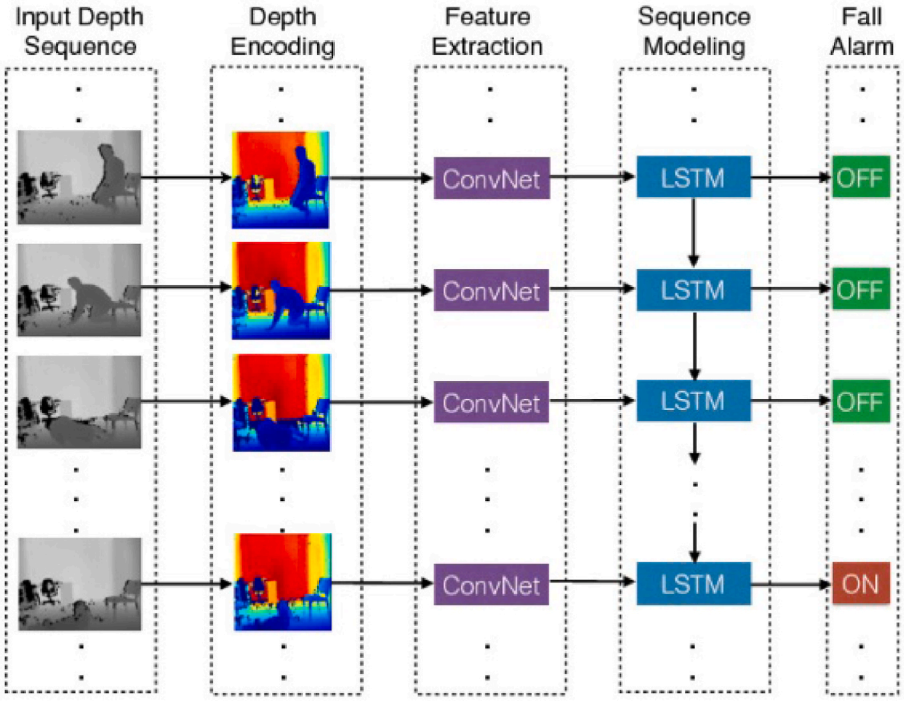


Fig. 19. Overview of the fall detection system as proposed by Abobakr et al. from [134].

Table 23
Results of the experiment reported by Lin et al. [143].

Metric	RNN	LSTM	GRU
Sensitivity (RP-Normalization)	93.7	100	96.4
Sensitivity (Linear Interpolation + RP-Normalization)	92.8	95.5	96.4
Specificity (RP-Normalization)	84.8	96.4	98.2
Specificity (Linear Interpolation + RP-Normalization)	87.5	94.6	98.2
Accuracy (RP-Normalization)	89.2	98.2	97.3
Accuracy (Linear Interpolation + RP-Normalization)	90.1	95	97.3

CNN followed by hybrid, LSTM, Auto-encoder and MLP as shown in Fig. 20.

These DL based techniques gives very good results, but some flaw is there. DL based techniques are in genral very data hungry and needs high computation especially for training of the model. It's very difficult to use these models in embedded IoT devices. In IoT devices not all computations is done in these devices locally, lots of files transfer from these devices (edges) to cloud servers and vice versa. This may be issue of security and privacy. The main focus of the works of the reviewed papers is the good results in terms of the metrics mentioned in section 3. Privacy and security should also be the focus of works. Also, majority of the works have been done using RGB inputs which is also a concern of the privacy of the subjects. To protect the privacy of the subjects more wokrs can be done using thermal and infrared sensors. Depth cameras use infrared to capture the subjects. These types of cameras can work in low light also. Here, there is no concern for privacy. Some works were done using multiple cameras ([8,78,121] etc.). The advantage of using multiple cameras is that even if one camera is not working or the subject

Table 24
Results as reported by Romaissa et al. [156].

Features	URFD				Le2i			
	Accur.	Precis.	Sens.	F Score	Accur.	Precis.	Sens.	F Score
Angle	0.960	1.000	0.8333	0.907	0.962	0.952	1.000	0.975
Angle + Distance	0.960	1.000	0.900	0.947	0.962	1.000	0.950	0.974
Average	0.960	1.000	0.86665	1.854	0.962	0.976	0.975	0.9745

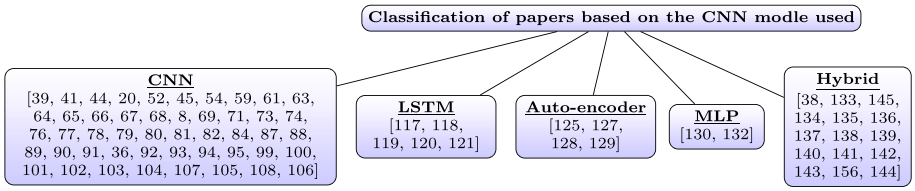


Fig. 20. Classification of papers based on the CNN model used.

is not in the view of both cameras then also the system will work with the available camera. Multiple cameras system is generally more complex and all cameras must work in a synchronize manner with each other.

7. Conclusion

In this paper, we have reviewed the recent (since 2014) developments in the DL based non-intrusive (vision-based) HFD methods. We have described the different metrics which are used to evaluate the performance of these fall detection methods. Sensitivity (recall), specificity, accuracy, F score, precision, and geometric mean were defined with their respective equations. The HFDSs which are publicly available, are also surveyed briefly. A brief description of MCFD, Le2i FDD, URFD, SDUFall, HQFSD, TSFD, SisFall, and UP-Fall datasets have been provided. To review the DL based fall detection methods, we have classified all methods according to the DL model used. We have classified it to CNN, Auto-Encoder, LSTM, MLP, and hybrid. Lots of work have been done in this field which gives good results. Some more work is needed considering the multi-person in a frame and occlusion. More work should be done considering privacy and security also. Thermal and infrared cameras can be utilized for privacy. Edge computing can be useful for providing security. The datasets which are available have been created by simulating the fall activities by some actors. These datasets do not contain the activities of the actual subjects (patients/old people). A dataset using thermal or infrared sensors can be created which will contain the activities of the actual subjects. Thermal and infrared sensors will also preserve the privacy of the subjects.

References

[1] World population ageing 2020 highlights - United Nat., https://www.un.org/development/desa/pd/sites/www.un.org.development.desa.pd/files/undesa_pd-2020_world_population_ageing_highlights.pdf, accessed: 2021-10-29 (2020).
[2] C. Rougier, J. Meunier, A. St-Arnaud, J. Rousseau, Monocular 3d head tracking to detect falls of elderly people, in: 2006 International Conference of the IEEE Engineering in Medicine and Biology Society, IEEE, 2006, pp. 6384–6387.
[3] D. Wild, U. Nayak, B. Isaacs, How dangerous are falls in old people at home? Br. Med. J. 282 (6260) (1981) 266–268.
[4] F. Marquis-Faulkes, S.J. McKenna, A.F. Newell, P. Gregor, Gathering the requirements for a fall monitor using drama and video with older people, Technol. Disabil. 17 (4) (2005) 227–236.
[5] S. Garfan, A. Alamoodi, B. Zaidan, M. Al-Zobbi, R.A. Hamid, J.K. Alwan, I. Y. Ahmaro, E.T. Khalid, F. Jumaah, O. Albahri, et al., Telehealth utilization during the covid-19 pandemic: a systematic review, Comput. Biol. Med. 138 (2021) 104878.
[6] A. Sufian, A. Ghosh, A.S. Sadiq, F. Smarandache, A survey on deep transfer learning to edge computing for mitigating the covid-19 pandemic, J. Syst. Architect. 108 (2020) 101830.
[7] A. Sufian, C. You, M. Dong, A deep transfer learning-based edge computing method for home health monitoring, in: 2021 55th Annual Conference on Information Sciences and Systems (CISS), IEEE, 2021, pp. 1–6.
[8] R. Espinosa, H. Ponce, S. Gutiérrez, L. Martínez-Villaseñor, J. Brieva, E. Moya-Albor, A vision-based approach for fall detection using multiple cameras and convolutional neural networks: a case study using the up-fall detection dataset, Comput. Biol. Med. 115 (2019) 103520.
[9] E. Alam, A. Sufian, A.K. Das, A. Bhattacharya, M.F. Ali, M.H. Rahman, Leveraging deep learning for computer vision: a review, in: 2021 22nd International Arab Conference on Information Technology (ACIT), IEEE, 2021, pp. 1–8.
[10] F. Sultana, A. Sufian, P. Dutta, Advancements in image classification using convolutional neural network, in: 2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), IEEE, 2018, pp. 122–129.
[11] B. Jena, S. Saxena, G.K. Nayak, L. Saba, N. Sharma, J.S. Suri, Artificial intelligence-based hybrid deep learning models for image classification: the first narrative review, Comput. Biol. Med. 137 (2021) 104803.
[12] A. Sufian, E. Alam, A. Ghosh, F. Sultana, D. De, M. Dong, Deep learning in computer vision through mobile edge computing for iot, in: Mobile Edge Computing, Springer, 2021, pp. 443–471.
[13] N. El-Bendary, Q. Tan, F. C. Pivot, A. Lam, Fall detection and prevention for the elderly: a review of trends and challenges., Int. J. Smart Sens. Intell. Syst. 6 (3).
[14] I. Putra, J. Brusey, E. Gaura, R. Vesilo, An event-triggered machine learning approach for accelerometer-based fall detection, Sensors 18 (1) (2018) 20.
[15] J. Gutiérrez, V. Rodríguez, S. Martin, Comprehensive review of vision-based fall detection systems, Sensors 21 (3) (2021) 947.
[16] X. Wang, J. Ellul, G. Azzopardi, Elderly fall detection systems: a literature survey, Front. Robotic. 7 (2020) 71.
[17] V.-R. Xefteris, A. Tsanousa, G. Meditskos, S. Vrochidis, I. Kompatsiaris, Performance, challenges, and limitations in multimodal fall detection systems: a review, IEEE Sensor. J..
[18] S. Rastogi, J. Singh, A Systematic Review on Machine Learning for Fall Detection System, Computational Intelligence.
[19] W. Zhu, N. Zeng, N. Wang, et al., Sensitivity, specificity, accuracy, associated confidence interval and roc analysis with practical sas implementations, in: NESUG Proceedings: Health Care and Life Sciences, vol. 19, 2010, p. 67. Baltimore, Maryland.
[20] X. Li, T. Pang, W. Liu, T. Wang, Fall detection for elderly person care using convolutional neural networks, in: 2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), IEEE, 2017, pp. 1–6.
[21] I. Charfi, J. Miteran, J. Dubois, M. Atri, R. Tourki, Optimized spatio-temporal descriptors for real-time fall detection: comparison of support vector machine and adaboost-based classification, J. Electron. Imag. 22 (4) (2013), 041106.
[22] E. Auvinet, C. Rougier, J. Meunier, A. St-Arnaud, J. Rousseau, Multiple cameras fall dataset, DIRO-Université de Montréal, Tech. Rep. 1350.
[23] B. Kwolek, M. Kepski, Human fall detection on embedded platform using depth maps and wireless accelerometer, Comput. Methods Progr. Biomed. 117 (3) (2014) 489–501.
[24] X. Ma, H. Wang, B. Xue, M. Zhou, B. Ji, Y. Li, Depth-based human fall detection via shape features and improved extreme learning machine, IEEE J. Biomed. Health Informatic. 18 (6) (2014) 1915–1922.
[25] A. Sucerquia, J.D. López, J.F. Vargas-Bonilla, Sisfall, A fall and movement dataset, Sensors 17 (1) (2017) 198.
[26] L. Martínez-Villaseñor, H. Ponce, J. Brieva, E. Moya-Albor, J. Núñez-Martínez, C. Peñafort-Asturiano, Up-fall detection dataset: a multimodal approach, Sensors 19 (9) (2019) 1988.
[27] S. Maldonado-Bascon, C. Iglesias-Iglesias, P. Martín-Martín, S. Lafuente-Arroyo, Fallen people detection capabilities using assistive robot, Electronics 8 (9) (2019) 915.
[28] C. Schuldt, I. Laptev, B. Caputo, Recognizing human actions: a local svm approach, in: Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR, vol. 3, IEEE, 2004, pp. 32–36, 2004.
[29] K. Soomro, A. R. Zamir, M. Shah, Ucf101: A Dataset of 101 Human Actions Classes from Videos in the Wild, arXiv preprint arXiv:1212.0402.
[30] J. Wang, Z. Liu, Y. Wu, J. Yuan, Mining actionlet ensemble for action recognition with depth cameras, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2012, pp. 1290–1297.
[31] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: a large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, Ieee, 2009, pp. 248–255.
[32] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C. L. Zitnick, Microsoft coco: common objects in context, in: European Conference on Computer Vision, Springer, 2014, pp. 740–755.
[33] J. Liu, A. Shahroudy, M. Perez, G. Wang, L.-Y. Duan, A.C. Kot, Ntu rgb+ d 120: a large-scale benchmark for 3d human activity understanding, IEEE Trans. Pattern Anal. Mach. Intell. 42 (10) (2019) 2684–2701.
[34] G. Baldewijns, G. Debard, G. Mertes, B. Vanrumste, T. Croonenborghs, Bridging the gap between real-life data and simulated data by providing a highly realistic fall dataset for evaluating camera-based fall detection algorithms, Healthcare Tech. Lett. 3 (1) (2016) 6–11.
[35] S. Vadivelu, S. Ganesan, O.R. Murthy, A. Dhall, Thermal imaging based elderly fall detection, in: Asian Conference on Computer Vision, Springer, 2016, pp. 541–553.
[36] L. A. Euprazia, K. Thyagharajan, A Novel Action Recognition System for Smart Monitoring of Elderly People Using Action Pattern Image and Series Cnn with Transfer Learning, arXiv preprint arXiv:2009.03285.

[37] M. Humenberger, S. Schraml, C. Sulzbachner, A.N. Belbachir, A. Srp, F. Vajda, Embedded fall detection with a neural network and bio-inspired stereo vision, in: 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, IEEE, 2012, pp. 60–67.

[38] P. Feng, M. Yu, S.M. Naqvi, J.A. Chambers, Deep learning for posture analysis in fall detection, in: 2014 19th International Conference on Digital Signal Processing, IEEE, 2014, pp. 12–17.

[39] N. Doulamis, Vision based fall detector exploiting deep learning, in: Proceedings of the 9th ACM International Conference on PErvasive Technologies Related to Assistive Environments, 2016, pp. 1–8.

[40] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, K.J. Lang, Phoneme recognition using time-delay neural networks, IEEE Trans. Acoust. Speech Signal Process. 37 (3) (1989) 328–339.

[41] Y. Fan, M.D. Levine, G. Wen, S. Qiu, A deep neural network for real-time detection of falling humans in naturally occurring scenes, Neurocomputing 260 (2017) 43–58.

[42] H. Bilen, B. Fernando, E. Gavves, A. Vedaldi, S. Gould, Dynamic image networks for action recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3034–3042.

[43] K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, arXiv preprint arXiv:1409.1556.

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

[45] Y.-Z. Hsieh, Y.-L. Jeng, Development of home intelligent fall detection iot system based on feedback optical flow convolutional neural network, IEEE Access 6 (2017) 6048–6057.

[46] B.K. Horn, B.G. Schunck, Determining optical flow, in: Techniques and Applications of Image Understanding, vol. 281, International Society for Optics and Photonics, 1981, pp. 319–331.

[47] S.S. Beauchemin, J.L. Barron, The computation of optical flow, ACM Comput. Surv. 27 (3) (1995) 433–466.

[48] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (10) (2009) 1345–1359.

[49] L. Torrey, J. Shavlik, Transfer learning, in: Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques, IGI global, 2010, pp. 242–264.

[50] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.

[51] A. Vedaldi, K. Lenc, Matconvnet: convolutional neural networks for matlab, in: Proceedings of the 23rd ACM International Conference on Multimedia, 2015, pp. 689–692.

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

[53] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A.Y. Ng, Ros: an open-source robot operating system, in: ICRA Workshop on Open Source Software, vol. 3, Kobe, Japan, 2009, p. 5.

[54] C. Iuga, P. Drăgan, L. Buşoniu, Fall monitoring and detection for at-risk persons using a uav, IFAC-PapersOnLine 51 (10) (2018) 199–204.

[55] P. Fahlstrom, T. Gleason, Introduction to UAV Systems, John Wiley & Sons, 2012.

[56] A. Zeggada, F. Melgani, Y. Bazi, A deep learning approach to uav image multilabeling, Geosci. Rem. Sens. Lett. IEEE 14 (5) (2017) 694–698.

[57] A. Hernandez, C. Copot, R. De Keyser, T. Vlas, I. Nascu, Identification and path following control of an ar. drone quadrotor, in: 2013 17th International Conference on System Theory, Control and Computing (ICSTCC), IEEE, 2013, pp. 583–588.

[58] J. Redmon, A. Farhadi, Yolo9000: better, faster, stronger, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 7263–7271.

[59] Z. Zhang, X. Ma, H. Wu, Y. Li, Fall detection in videos with trajectory-weighted deep-convolutional rank-pooling descriptor, IEEE Access 7 (2018) 4135–4144.

[60] L. Wang, Y. Qiao, X. Tang, Action recognition with trajectory-pooled deep-convolutional descriptors, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 4305–4314.

[61] L. Shen, Q. Zhang, G. Cao, H. Xu, Fall detection system based on deep learning and image processing in cloud environment, in: Conference on Complex, Intelligent, and Software Intensive Systems, Springer, 2018, pp. 590–598.

[62] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P.V. Gehler, B. Schiele, Deepcut: joint subset partition and labeling for multi person pose estimation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4929–4937.

[63] X. Kong, L. Chen, Z. Wang, Y. Chen, L. Meng, H. Tomiyama, Robust self-adaptation fall-detection system based on camera height, Sensors 19 (17) (2019) 3768.

[64] A. El Kaid, K. Baïna, J. Baïna, Reduce false positive alerts for elderly person fall video-detection algorithm by convolutional neural network model, Procedia Comput. Sci. 148 (2019) 2–11.

[65] S.A. Cameiro, G.P. da Silva, G.V. Leite, R. Moreno, S.J.F. Guimarães, H. Pedrini, Multi-stream deep convolutional network using high-level features applied to fall detection in video sequences, in: 2019 International Conference on Systems, Signals and Image Processing (IWSSIP), IEEE, 2019, pp. 293–298.

[66] G. Leite, G. Silva, H. Pedrini, Fall detection in video sequences based on a three-stream convolutional neural network, in: 2019 18th IEEE International Conference on Machine Learning and Applications (ICMLA), IEEE, 2019, pp. 191–195.

[67] J. Brieva, H. Ponce, E. Moya-Albor, L. Martínez-Villaseñor, An intelligent human fall detection system using a vision-based strategy, in: 2019 IEEE 14th International Symposium on Autonomous Decentralized System (ISADS), IEEE, 2019, pp. 1–5.

[68] X. Cai, S. Li, X. Liu, G. Han, A novel method based on optical flow combining with wide residual network for fall detection, in: 2019 IEEE 19th International Conference on Communication Technology (ICCT), IEEE, 2019, pp. 715–718.

[69] X. Cai, X. Liu, S. Li, G. Han, Fall detection based on colorization coded mhi combining with convolutional neural network, in: 2019 IEEE 19th International Conference on Communication Technology (ICCT), IEEE, 2019, pp. 1694–1698.

[70] A.F. Bobick, J.W. Davis, The recognition of human movement using temporal templates, IEEE Trans. Pattern Anal. Mach. Intell. 23 (3) (2001) 257–267.

[71] S. Kasturi, A. Filonenko, K.-H. Jo, Human fall recognition using the spatiotemporal 3d cnn, in: Proc. IW-FCV, 2019, pp. 1–3.

[72] D. Tran, L. Bourdev, R. Fergus, L. Torresani, M. Paluri, Learning spatiotemporal features with 3d convolutional networks, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 4489–4497.

[73] J. Wu, K. Wang, B. Cheng, R. Li, C. Chen, T. Zhou, Skeleton based fall detection with convolutional neural network, in: 2019 Chinese Control and Decision Conference (CCDC), IEEE, 2019, pp. 5266–5271.

[74] Y. Zheng, D. Zhang, L. Yang, Z. Zhou, Fall detection and recognition based on gcn and 2d pose, in: 2019 6th International Conference on Systems and Informatics (ICSAI), IEEE, 2019, pp. 558–562.

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

[76] R. Espinosa, H. Ponce, S. Gutiérrez, L. Martínez-Villaseñor, J. Brieva, E. Moya-Albor, Application of convolutional neural networks for fall detection using multiple cameras, in: Challenges and Trends in Multimodal Fall Detection for Healthcare, Springer, 2020, pp. 97–120.

[77] A. Carlier, P. Peyramaure, K. Favre, M. Pressigout, Fall detector adapted to nursing home needs through an optical-flow based cnn, in: 2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), IEEE, 2020, pp. 5741–5744.

[78] C. Yao, J. Hu, W. Min, Z. Deng, S. Zou, W. Min, A novel real-time fall detection method based on head segmentation and convolutional neural network, J. Real Time. Image. Process. 17 (2020) 1939–1949.

[79] C. Menacho, J. Ordoñez, Fall detection based on cnn models implemented on a mobile robot, in: 2020 17th International Conference on Ubiquitous Robots (UR), IEEE, 2020, pp. 284–289.

[80] Y. Chen, X. Kong, L. Meng, H. Tomiyama, An edge computing based fall detection system for elderly persons, Procedia Comput. Sci. 174 (2020) 9–14.

[81] E.P. Ijjina, Human Fall Detection Using Temporal Templates and Convolutional Neural Networks, ICCII, 2020, p. 763, 2018.

[82] G.K. Hader, M.M. Ben Ismail, O. Bchir, Automatic fall detection using region-based convolutional neural network, Int. J. Inj. Control Saf. Promot. 27 (4) (2020) 546–557.

[83] S. Ren, K. He, R. Girshick, J. Sun, Faster r-cnn: towards real-time object detection with region proposal networks, Adv. Neural Inf. Process. Syst. 28 (2015) 91–99.

[84] R. Dichwalkar, S. Oak, T. Rajabally, D. Kalbande, Activity recognition and fall detection in elderly people, in: 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), IEEE, 2020, pp. 1–6.

[85] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, Mobilenets: Efficient Convolutional Neural Networks for Mobile Vision Applications, arXiv preprint arXiv:1704.04861.

[86] S.N. Robinovitch, F. Feldman, Y. Yang, R. Schonnop, P.M. Leung, T. Sarraf, J. Sims-Gould, M. Loughin, Video capture of the circumstances of falls in elderly people residing in long-term care: an observational study, Lancet 381 (9860) (2013) 47–54.

[87] U. Asif, S. Von Cavallar, J. Tang, S. Harrer, Sshfd: Single Shot Human Fall Detection with Occluded Joints Resilience, arXiv preprint arXiv:2004.00797.

[88] F. Lezzar, D. Benmerzoug, I. Kitouni, Camera-based fall detection system for the elderly with occlusion recognition, Appl. Med. Informatic. 42 (3) (2020) 169–179.

[89] J. Zhang, C. Wu, Y. Wang, Human fall detection based on body posture spatio-temporal evolution, Sensors 20 (3) (2020) 946.

[90] C. Zhong, W.W. Ng, S. Zhang, C.D. Nugent, C. Shewell, J. Medina-Quero, Multi-occupancy fall detection using non-invasive thermal vision sensor, IEEE Sensor. J. 21 (4) (2020) 5377–5388.

[91] U. Asif, B. Mashford, S. Von Cavallar, S. Yohanandan, S. Roy, J. Tang, S. Harrer, Privacy preserving human fall detection using video data, in: Machine Learning for Health Workshop, 2020, pp. 39–51.

[92] Y. Liu, Y. Deng, C. Jia, Y. Yang, R. Wang, C. Li, Two-stream Graph Convolutional Networks for 2d Skeleton-Based Fall Detection.

[93] W. Chen, Z. Jiang, H. Guo, X. Ni, Fall detection based on key points of human-skeleton using openpose, Symmetry 12 (5) (2020) 744.

[94] I. Kareem, S.F. Ali, A. Sheharyar, Using skeleton based optimized residual neural network architecture of deep learning for human fall detection, in: 2020 IEEE 23rd International Multitopic Conference (INMIC), IEEE, 2020, pp. 1–5.

[95] H. Abdo, K.M. Amin, A.M. Hamad, Fall detection based on retinanet and mobilenet convolutional neural networks, in: 2020 15th International Conference on Computer Engineering and Systems (ICCES), IEEE, 2020, pp. 1–7.

[96] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2980–2988.

[97] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, Automatic Differentiation in Pytorch.

[98] J.W.D.L. Sijie Yan, Yuanjun Xiong, MMSkeleton. https://github.com/open-mmlab/mmskeleton, 2019, 2021-09-15.

[99] S. Chhetri, A. Alsadoon, T. Al-Dala'in, P. Prasad, T.A. Rashid, A. Maag, Deep learning for vision-based fall detection system: enhanced optical dynamic flow, Comput. Intell. 37 (1) (2021) 578–595.

[100] Z. Chen, Y. Wang, W. Yang, Video Based Fall Detection Using Human Poses, arXiv preprint arXiv:2107.14633.

[101] L. Killian, M. Julien, B. Kevin, L. Maxime, B. Carolina, C. Mélanie, B. Nathalie, G. Sylvain, G. Sebastien, Fall prevention and detection in smart homes using monocular cameras and an interactive social robot, in: Proceedings of the Conference on Information Technology for Social Good, 2021, pp. 7–12.

[102] G.V. Leite, G.P. da Silva, H. Pedrini, Three-stream convolutional neural network for human fall detection, in: Deep Learning Applications, vol. 2, Springer, 2021, pp. 49–80.

[103] S. Zou, W. Min, L. Liu, Q. Wang, X. Zhou, Movement tube detection network integrating 3d cnn and object detection framework to detect fall, Electronics 10 (8) (2021) 898.

[104] C. Vishnu, R. Datla, D. Roy, S. Babu, C. K. Mohan, Human fall detection in surveillance videos using fall motion vector modeling, IEEE Sensor. J..

[105] X. Cai, X. Liu, M. An, G. Han, Vision-based fall detection using dense block with multi-channel convolutional fusion strategy, IEEE Access 9 (2021) 18318–18325.

[106] O. Keskes, R. Noumeir, Vision-based fall detection using st-gcn, IEEE Access 9 (2021) 28224–28236.

[107] S.J. Berlin, M. John, Vision based human fall detection with siamese convolutional neural networks, J. Ambient Intell. Hum. Comput. (2021) 1–12.

[108] H. Li, C. Li, Y. Ding, Fall detection based on fused saliency maps, Multimed. Tool. Appl. 80 (2) (2021) 1883–1900.

[109] D. Gouaillier, V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcade, B. Marnier, J. Serre, B. Maisonnier, Mechatronic design of nao humanoid, in: 2009 IEEE International Conference on Robotics and Automation, IEEE, 2009, pp. 769–774.

[110] I. Charfi, J. Miteran, J. Dubois, M. Atri, R. Tourki, Definition and performance evaluation of a robust svm based fall detection solution, in: 2012 Eighth International Conference on Signal Image Technology and Internet Based Systems, IEEE, 2012, pp. 218–224.

[111] G. Koch, R. Zemel, R. Salakhutdinov, et al., Siamese neural networks for one-shot image recognition, in: ICML Deep Learning Workshop, vol. 2, Lille, 2015.

[112] A. He, C. Luo, X. Tian, W. Zeng, A twofold siamese network for real-time object tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4834–4843.

[113] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al., Matching networks for one shot learning, Adv. Neural Inf. Process. Syst. 29 (2016) 3630–3638.

[114] S. Yan, Y. Xiong, D. Lin, Spatial temporal graph convolutional networks for skeleton-based action recognition, in: Thirty-second AAAI Conference on Artificial Intelligence, 2018.

[115] E. Cippitelli, E. Gambi, S. Gasparrini, S. Spinsante, Tst fall detection dataset v2, IEEE Dataport, IEEE. https://doi. org/10.21227/H2VC7J.

[116] M.S. Alzahrani, S.K. Jarraya, M.A. Salamah, H. Ben-Abdallah, Fallfree, Multiple fall scenario dataset of cane users for monitoring applications using kinect, in: 2017 13th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), IEEE, 2017, pp. 327–333.

[117] M.M. Hasan, M.S. Islam, S. Abdullah, Robust pose-based human fall detection using recurrent neural network, in: 2019 IEEE International Conference on Robotics, Automation, Artificial-Intelligence and Internet-Of-Things (RAAICON), IEEE, 2019, pp. 48–51.

[118] S. Jeong, S. Kang, I. 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), IEEE, 2019, pp. 1–4.

[119] Q. Feng, C. Gao, L. Wang, Y. Zhao, T. Song, Q. Li, Spatio-temporal fall event detection in complex scenes using attention guided lstm, Pattern Recogn. Lett. 130 (2020) 242–249.

[120] B.D. Romaissa, O. Mourad, N. Brahim, B. Yazid, Fall detection using body geometry in video sequences, in: 2020 Tenth International Conference on Image Processing Theory, Tools and Applications (IPTA), IEEE, 2020, pp. 1–5.

[121] M. Taufeeque, S. Koita, N. Spicher, T.M. Deserno, Multi-camera, multi-person, and real-time fall detection using long short term memory, in: Medical Imaging 2021: Imaging Informatics for Healthcare, Research, and Applications, vol. 11601, 2021, p. 1160109. International Society for Optics and Photonics.

[122] N. Wojke, A. Bewley, D. Paulus, Simple online and realtime tracking with a deep association metric, in: 2017 IEEE International Conference on Image Processing (ICIP), IEEE, 2017, pp. 3645–3649.

[123] S. Kreiss, L. Bertoni, A. Alahi, Pifpaf: composite fields for human pose estimation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 11977–11986.

[124] D. Gale, L.S. Shapley, College admissions and the stability of marriage, Am. Math. Mon. 69 (1) (1962) 9–15.

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

[126] A. Mitseva, S. Kyriazakos, A. Litke, N. Papadakis, N. Prasad, Isisemd: intelligent system for independent living and self-care of seniors with mild cognitive impairment or mild dementia, J. Info. Tech. Healthcare. 7 (6) (2009) 383–399.

[127] J. Nogas, S.S. Khan, A. Mihailidis, Fall detection from thermal camera using convolutional lstm autoencoder, in: Proceedings of the 2nd Workshop on Aging, Rehabilitation and Independent Assisted Living, IJCAI Workshop, 2018.

[128] F. A. Elshwemy, R. Elbasiony, M. T. Saidahmed, A new approach for thermal vision based fall detection using residual autoencoder, Int. J. Intell. Eng. Syst 13 (2).

[129] J. Nogas, S.S. Khan, A. Mihailidis, Deepfall: non-invasive fall detection with deep spatio-temporal convolutional autoencoders, Journal of Healthcare Informatics Research 4 (1) (2020) 50–70.

[130] M. Safarzadeh, Y. Alborzi, A.N. Ardekany, Real-time fall detection and alert system using pose estimation, in: 2019 7th International Conference on Robotics and Mechatronics (ICRoM), IEEE, 2019, pp. 508–511.

[131] S.-E. Wei, V. Ramakrishna, T. Kanade, Y. Sheikh, Convolutional pose machines, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4724–4732.

[132] H. Ramirez, S.A. Velastin, I. Meza, E. Fabregas, D. Makris, G. Farias, Fall detection and activity recognition using human skeleton features, IEEE Access 9 (2021) 33532–33542.

[133] N. Lu, X. Ren, J. Song, Y. Wu, Visual guided deep learning scheme for fall detection, in: 2017 13th IEEE Conference on Automation Science and Engineering (CASE), IEEE, 2017, pp. 801–806.

[134] A. Abobakr, M. Hossny, H. Abdelkader, S. Nahavandi, Rgb-d fall detection via deep residual convolutional lstm networks, in: 2018 Digital Image Computing: Techniques and Applications (DICTA), IEEE, 2018, pp. 1–7.

[135] C. Ma, A. Shimada, H. Uchiyama, H. Nagahara, R.-i. Taniguchi, Fall detection using optical level anonymous image sensing system, Opt Laser. Technol. 110 (2019) 44–61.

[136] J. Zhou, T. Komuro, Recognizing fall actions from videos using reconstruction error of variational autoencoder, in: 2019 IEEE International Conference on Image Processing (ICIP), IEEE, 2019, pp. 3372–3376.

[137] X. Cai, S. Li, X. Liu, G. Han, Vision-based fall detection with multi-task hourglass convolutional auto-encoder, IEEE Access 8 (2020) 44493–44502.

[138] R. Manekar, S. Saurav, S. Maiti, S. Singh, S. Chaudhury, R. Kumar, K. Chaudhary, et al., Activity recognition for indoor fall detection in 360-degree videos using deep learning techniques, in: Proceedings of 3rd International Conference on Computer Vision and Image Processing, Springer, 2020, pp. 417–429.

[139] Y. Chen, W. Li, L. Wang, J. Hu, M. Ye, Vision-based fall event detection in complex background using attention guided bi-directional lstm, IEEE Access 8 (2020) 161337–161348.

[140] M.T. Pourazad, A. Shojaei-Hashemi, P. Nasiopoulos, M. Azimi, M. Mak, J. Grace, D. Jung, T. Bains, A non-intrusive deep learning based fall detection scheme using video cameras, in: 2020 International Conference on Information Networking (ICOIN), IEEE, 2020, pp. 443–446.

[141] J. Li, S.-T. Xia, Q. Ding, Multi-level recognition on falls from activities of daily living, in: Proceedings of the 2020 International Conference on Multimedia Retrieval, 2020, pp. 464–471.

[142] V. Mehta, A. Dhall, S. Pal, S.S. Khan, Motion and region aware adversarial learning for fall detection with thermal imaging, in: 2020 25th International Conference on Pattern Recognition (ICPR), IEEE, 2021, pp. 6321–6328.

[143] C.-B. Lin, Z. Dong, W.-K. Kuan, Y.-F. Huang, A framework for fall detection based on openpose skeleton and lstm/gru models, Appl. Sci. 11 (1) (2021) 329.

[144] A. Apicella, L. Snidaro, Deep neural networks for real-time remote fall detection, in: International Conference on Pattern Recognition, Springer, 2021, pp. 188–201.

[145] N. Lu, Y. Wu, L. Feng, J. Song, Deep learning for fall detection: three-dimensional cnn combined with lstm on video kinematic data, IEEE J. Biomed. Health Informatic. 23 (1) (2018) 314–323.

[146] Z. Zhang, Microsoft kinect sensor and its effect, IEEE Multimedia 19 (2) (2012) 4–10.

[147] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[148] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.

[149] Y. Zhang, Y. Lu, H. Nagahara, R.-i. Taniguchi, Anonymous camera for privacy protection, in: 2014 22nd International Conference on Pattern Recognition, IEEE, 2014, pp. 4170–4175.

[150] D. P. Kingma, M. Welling, Auto-encoding Variational Bayes, arXiv preprint arXiv: 1312.6114.

[151] K. Hara, H. Kataoka, Y. Satoh, Learning spatio-temporal features with 3d residual networks for action recognition, in: Proceedings of the IEEE International Conference on Computer Vision Workshops, 2017, pp. 3154–3160.

[152] H.-S. Fang, S. Xie, Y.-W. Tai, C. Lu, Rmpe: regional multi-person pose estimation, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2334–2343.

[153] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn, IEEE Trans. Pattern Anal. Mach. Intell. 42 (2) (2020) 386–397, https://doi.org/10.1109/TPAMI.2018.2844175.
[154] J. Dai, Y. Li, K. He, J. Sun, R-fcn, Object detection via region-based fully convolutional networks, in: Advances in Neural Information Processing Systems, 2016, pp. 379–387.
[155] N. Otsu, A threshold selection method from gray-level histograms, IEEE Trans. Syst. Man Cybernet. 9 (1) (1979) 62–66.
[156] B.D. Romaissa, O. Mourad, N. Brahim, B. Yazid, Vision-based fall detection using body geometry, in: International Conference on Pattern Recognition, Springer, 2021, pp. 170–185.
[157] G. Papandreou, T. Zhu, L.-C. Chen, S. Gidaris, J. Tompson, K. Murphy, Personlab: person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model, in: Proceedings of the European Conference on Computer Vision, ECCV, 2018, pp. 269–286.