Real-Time Fall Detection with Contextual Awareness: A Deep Learning Approach to Mitigating False Alarms

Fouzi Lezzar[1], Seif Eddine Mili[2], Nabila Azeri[3] and Djamel Benmerzoug[1]

[1]LIRE Laboratory, Faculty of New Technologies of Information and Communication, University of Abdelhamid Mehri Constantine 2, Constantine, Algeria
[2]Ecole Normale Supérieure, Constantine, LISCO laboratory, Annaba University, Algeria
[3]Mathematics and Informatics Department, Abbes Laghrour University, Khenchela, Algeria
fouzi.lezzar@univ-constantine2.dz

Abstract: Falls among elderly individuals pose a serious health risk, often resulting in severe injuries or death, yet existing vision-based detection systems suffer from high false alarm rates due to their reliance on posture estimation alone, without considering environmental context. To address this limitation, we propose a novel, context-aware deep learning system that integrates posture recognition with environmental analysis to improve fall detection accuracy. Our approach leverages a fine-tuned YOLOv10 model and a custom CNN architecture for real-time posture detection and contextual classification, enhanced by multi-dataset training and occlusion-aware strategies. Evaluated across multiple datasets, the system demonstrates robust performance under varying lighting, occlusions, and complex backgrounds, achieving high accuracy, precision, recall, and F1-score while significantly reducing false positives. These results highlight its potential as a reliable, real-time solution for practical deployment in elderly care settings, offering a scalable and efficient alternative to existing methods.

Keywords: Fall detection; Deep Learning; Transfer Learning; YOLOv10; CNN; Person's environment; Occlusions

1. Introduction

Falls are a significant indicator of frailty, immobility, and both acute and chronic health issues in elderly individuals. They are among the leading causes of injury-related deaths world wide, particularly among older adults living alone. According to the World Health Organization (WHO) [1], approximately 30% of individuals over the age of 65 experience at least one fall per year, a percentage that increases with age. The National Council on Aging (NCOA) [2] reports that in the United States, an older adult dies from a fall-related injury every 19 minutes, resulting in 27000 deaths annually. Beyond fatal incidents, falls often lead to serious complications such as fractures, reduced mobility, and loss of independence, significantly affecting the quality of life of elderly individuals. Given these alarming statistics, developing effective and reliable fall detection systems is crucial to ensure rapid intervention and minimize health risks.

Several fall detection systems have been proposed in the literature and are generally categorized into wearable-based, ambient sensor-based, and vision-based methods [3][4][5]. Wearable sensors, such as accelerometers and gyroscopes, have gained popularity due to their portability and low cost [6]. However, these solutions require continuous user compliance, which is often impractical for elderly individuals. Ambient sensor-based solutions, including radar- and microphone-based systems, eliminate the need for user participation but are highly sensitive to environmental noise and occlusions, limiting their reliability in real-world applications [6][7].

Among these approaches, vision-based methods using deep learning have demonstrated significant potential for automated and non-intrusive fall detection. However, they face several challenges, including lighting variations, body posture changes, occlusions, and background clutter [4][5][9]. One of the most critical limitations of existing vision-based solutions is the difficulty in distinguishing between a person lying on the floor due to a fall and someone lying

Received: May 31th, 2025. Accepted: December 28th, 2025
DOI: 10.15676/ijeei.2025.17.4.12

on a bed or couch [10]. Many models rely solely on body posture estimation while ignoring environmental context, resulting in high false positive rates. For instance, a person lying horizontally in an image may be misclassified as having fallen, even if they are resting on a mattress. Furthermore, occlusions caused by furniture or other objects pose additional challenges, as many detection systems struggle to recognize partially visible individuals. These limitations highlight the need for a more robust and context-aware fall detection system.

To address these challenges, we propose a context-aware deep learning-based fall detection system that integrates posture recognition with environmental context analysis to improve classification accuracy. Our approach leverages a fine-tuned YOLOv10 model and a custom CNN architecture, enabling real-time detection while reducing false alarms. Unlike traditional methods, the proposed system not only analyses body posture but also incorporates spatial and environmental cues, allowing it to distinguish between actual falls and non-fall situations. Additionally, robustness to occlusions is enhanced through data augmentation techniques and an optimized annotation strategy.

The main contributions of this work are summarized as follows:

1. A novel deep learning-based approach that integrates fall posture recognition with environmental context awareness, significantly reducing false positives.
2. A fine-tuned YOLOv10 model and a custom CNN architecture optimized for real-time performance in complex scenarios.
3. A multi-dataset training strategy that combines public and locally collected datasets to enhance model generalization across diverse environments using context-specific annotation techniques.
4. An occlusion-aware detection strategy that ensures reliable fall classification even when individuals are partially obscured by objects.

The remainder of this paper is organized as follows: Section 2 reviews related fall detection methods. Section 3 describes the proposed approach, including dataset details, pre-processing steps, and model architecture. Section 4 presents the experimental results, while Section 5 discusses an ablation study. Section 6 provides a comparative analysis with existing solutions. Finally, Section 7 concludes the paper and outlines future research directions.

2. Related work

Many types of fall detection systems have been developed over the years. Existing systems can be divided into three categories: wearable-based, camera-based and ambient-based fall detection systems [3][4][5].

A. Wearable-based Systems

Wearable sensors for fall detection systems are integrated into devices worn by the subject. These devices use accelerometers, gyroscopes, glucometers, pressure sensors, ECG, EEG, EOG. Wearable devices have been identified as an important future technology for fall detection due to their mobility, portability, low cost, and ease of use. For example, in [11], a deep learning model combining a Temporal Convolutional Network (TCN) and a Gated Recurrent Unit (GRU) was proposed to extract high-level features for classification. To collect data for training and evaluation, an Inertial Measurement Unit (IMU) was used. In another study [12], researchers developed a battery-free human motion sensing system capable of recognizing movements through a shoe-embedded triboelectric nanogenerator (TENG). For signal collection, this system utilized a conductive PVA-PEDOT: PSS hydrogel. Furthermore, an artificial intelligence (AI)-based fall detection system leveraging TENG technology was implemented. A different study [13] employed a small Convolutional Neural Network, also known as TinyCNN, featuring a two-stage process for extracting relevant characteristics. The model was tested on two public fall detection databases, KFall and SisFall. Another approach to fall detection is outlined in [14], where researchers built software architecture based on a Recurrent Neural Network (RNN). The study in [15] enables fall detection to be performed

entirely on the wearable device without requiring external servers. This study used common devices such as a smartphone, Raspberry Pi, Arduino, and NodeMCU to detect fall situations. An accelerometer was placed inside user's left or right pants pockets to continuously transmit data to a multithreaded server. The server used a pre-trained machine learning (ML) model to analyse the data and identify falls. If a fall was detected, a message containing the user's location was sent to a designated contact. In another study [16], the authors evaluated the potential of a personalized model that incorporated deep neural networks and ensemble learning techniques to improve fall detection performance. Additionally, in [17], a cross-platform wearable device for gait analysis and fall detection was proposed. The system operated through a web application, making it independent of any specific platform. Fall detection was performed in real time using an SVM classifier. Lastly, authors in [18] compared two deep learning-based methods for differentiating between fall and non-fall activities. The researchers utilized signals from wearable sensors to train and evaluate two models: one based on Long Short-Term Memory (LSTM) networks and another using Transformer encoders.

B. Vision-based Systems

Vision-based fall detection systems rely on image processing techniques applied to video frames or images captured by cameras placed around the Region of Interest (ROI). Their accuracy can be improved by incorporating machine learning algorithms alongside image-processing techniques. In [19], the authors proposed a two-stream network named Flow-Pose Net (FP-Net). This approach combines optical flow and human pose information to achieve robust and accurate fall detection in videos. A human pose estimator was used to identify body joint locations and design a Graph Convolutional Network (GCN)-based model to extract body appearance features from the detected poses. Additionally, the authors employed a CNN for feature extraction. In another study [20], a Multimodal Spatiotemporal Skeletal Kinematic Gait Feature Fusion (MSTSK-GFF) classifier was proposed for fall detection in videos. This method utilizes a multimodal feature fusion learning approach, incorporating a Spatiotemporal Graph Convolutional Network (STGCN) for skeletal modelling and 1D-CNNs for temporal gait modelling to generate two sets of spatiotemporal kinematic gait features from the skeleton frame sequences. These feature sets are then fused through a concatenation process to develop the final classification model. To address data limitations and computational efficiency challenges, another study [21] applied transfer learning by fine-tuning a pre-trained model on a large-scale image dataset for fall detection. The proposed approach achieved a test accuracy of 98.15%. In [22], an efficient activity recognition and fall detection system (ARFDNet) was introduced. A pose estimation network was used to extract skeletal features from raw RGB video. The skeleton coordinates were then processed by a CNN, followed by Gated Recurrent Units (GRUs) and fully connected layers for classification. The study in [23] tackled the issue of high false-positive rates by proposing various topologies of a multimodal CNN trained to detect falls using both RGB images and accelerometer data. The proposed system was evaluated on the UR Fall and UP-Fall datasets. In [24], fall detection was performed by analysing human body geometry across different frames in a video sequence. Pose estimation techniques were used to compute the angle and distance between the vector formed by the head-centroid of the detected facial image and the body's centre hip, as well as the vector aligned with the horizontal axis of the centre hip, to create distinctive image features for classification. Furthermore, a fall detection method utilizing input-level data fusion was proposed in [25]. This method integrates 1D time-series data and 2D visual data to enhance information complementarity. The data was collected from wearable devices and cameras. The authors of [26] introduced an enhanced version of YOLOv5s, named YOLOv5s-GCC, specifically designed to detect falls in elderly individuals. Finally, a novel fall detection approach was proposed in [27], relying solely on images captured by a standard video camera, eliminating the need for environmental sensors. This method utilizes feature extraction based on human skeleton estimation.

C. *Ambient-based Systems*

In addition to wearable-based and camera-based systems, some other solutions that have not been extensively documented in the literature are based on ambient devices. These systems operate using sensors placed in the environment. Sensors such as active infrared, RFID, pressure sensors, smart tiles, magnetic switches, Doppler radar, ultrasonic sensors, and microphones are employed to detect environmental changes caused by falls. For example, in [28], millimetre-wave (mmWave) radar technology was used for a non-intrusive human fall detection solution. Various classifiers, such as Random Forest (RF) and Support Vector Machine (SVM), were applied in the classification process using features extracted from these devices. In another study [29], the authors recorded reflected signals using a microphone, capturing the resulting Doppler shift when a fall occurred.

3. Proposed solution

The proposed approach addresses a critical limitation of camera-based fall detection systems. Specifically, most existing solutions are unable to reliably distinguish between a person lying on the floor due to a fall and a person lying on a bed, which can cause confusion and false detections. Following the presentation of the proposed approach, we describe the datasets used in this study, as well as the experiments and evaluation procedures conducted.

A. *Principle of the Solution*

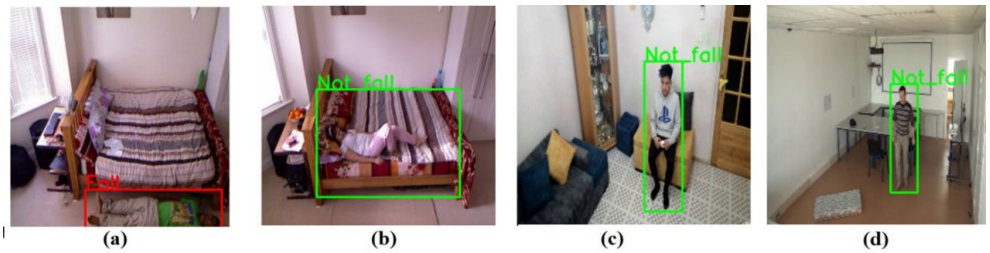


Figure 1. Different considered postures of a person.

Our approach focuses on both the posture of the person and the surrounding environment or context, which plays a crucial role in verifying a fall situation. A clear way to illustrate this principle is through the example of a person lying on a bed versus lying on the ground. The system must not only analyse the person's posture but also consider their surroundings, as posture alone is insufficient to declare an emergency fall. It is evident that a person lying on a bed and a person lying on the ground represent two distinct situations with clearly different environments. Ignoring this distinction, as seen in many existing solutions in the literature, can lead to misclassification. Another frequently overlooked challenge in the literature is occlusion, which this paper also aims to address.

1. In Fig. 1(a), the system detects a person lying on the ground as being in a fall situation.
2. In Fig. 1(b), despite the person being in a lying position, he is not classified as having fallen since he is on a bed.
3. Fig. 1(c) and (d) depict ordinary, non-fall situations: sitting and walking, respectively.

This fall detection system adopts a unified deep learning approach that simultaneously identifies a person's posture while considering their surrounding environment, addressing a key limitation in existing solutions. Traditional methods rely on multi-step detection pipelines, typically using a convolutional neural network (CNN) for person detection followed by a separate machine learning classifier for fall recognition. This multi-stage process increases

computational complexity and error propagation due to accumulated inaccuracies from individual models.

To overcome these challenges, we propose a single-step fall detection framework. A dataset is prepared and used to train the two proposed architectures: a CNN model with four convolutional layers and a transfer learning-based pre-trained YOLOv10 model [30]. These models are designed for classification, categorizing postures into two classes: Fall and Non Fall. The proposed approach is evaluated using multiple performance metrics, including accuracy, precision, recall, and F1-score. Once trained, the model is deployed with a camera (Fig. 2), as follows:

Repeat

1- **Capture Frame:** The camera captures an image (video is considered as a sequence of frames).
2- **Image Processing:** The system processes the image to detect a fall.
3- **Fall Detection:**

- If a fall is detected, then counter C is incremented.
- If the counter exceeds a predefined threshold ***T***, an alert is triggered. The alert and the captured image containing the fall action are sent to the responsible person. Real-time access to the camera is available to caregivers for further verification.

4- **Non-Fall Detection:** If no fall is detected, then reset the counter to zero.

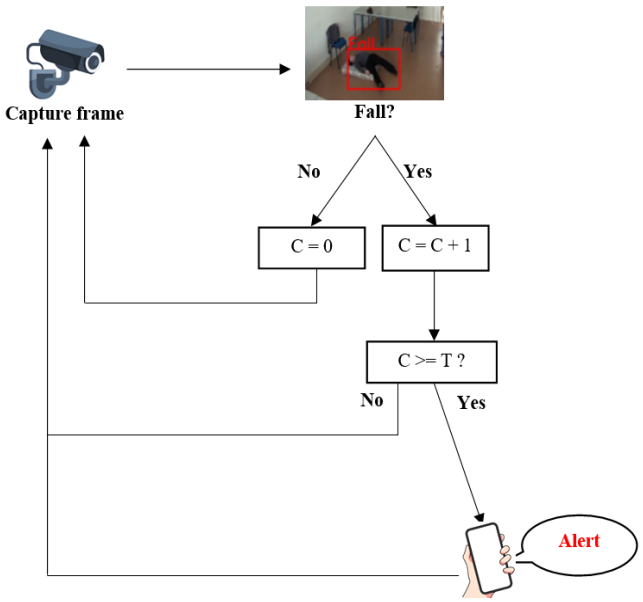


Figure 2. Fall detection process

To enhance detection reliability and reduce false alarms caused by transient movements or brief occlusions, the proposed system incorporates a temporal validation mechanism based on a persistence counter C and a predefined threshold T. Although fall classification is performed on individual frames, the alerting decision relies on the continuity of the detected fall state across consecutive frames. Specifically, when a fall is detected, the counter C is incremented. An emergency alert is triggered only if the fall state persists over time such that $C \geq T$. Conversely, if a subsequent frame is classified as Not Fall, the counter is immediately reset to zero. By calibrating the threshold T according to the camera frame rate, the system ensures that alerts are generated based on sustained events rather than isolated detections, thereby improving robustness and reducing false alarms.

The common challenges encountered in the development of fall detection systems, as outlined in the introduction, will be addressed in this paper as follows:

1. Variations In Lighting, Scale, Object Deformation, Background Noise, And DifferencesiIn Body Size: These Challenges Will Be Mitigated By Using A Diverse Dataset and Selecting an Optimal Deep Learning Model.
2. Occlusions, Diverse Orientations and Positions of A Fallen Individual, And Distinguishing Between A Person Lying On The Floor Due To A Fall and One Lying on A Bed or Mattress: These issues will be tackled through an appropriate annotation technique and the careful selection of a deep learning model.

B. Dataset Description

In this study, we used two public datasets (Le2i [31] and the Fall Detection Dataset [32]) for training, validation, and testing, along with a third dataset that we developed locally for additional testing.

The Le2i Fall Dataset (Fig. 1d) consists of 124 RGB videos, including 94 fall videos and 30 ADL (Activities of Daily Living) videos, recorded in two different simulated environments. It captures falls from various angles (forward, backward, and sideways) and from different initial positions, such as standing and sitting. The dataset also includes videos with varying lighting conditions and occlusions. The videos were recorded using a single camera positioned 2.4 meters high, with a resolution of 320×240 pixels and a frame rate of 25 fps. To integrate this dataset with the second one, we extracted frames from the videos.

The Fall Detection Dataset (Fig. 1a and 1b) contains RGB images resized to 320×240 pixels. The original videos were recorded at a 640×480 resolution using a single Kinect sensor mounted at approximately 2.4 meters in height. The dataset comprises 22636 images, though not all were suitable for training. The images were captured in various rooms, from different angles, and feature multiple individuals of varying body sizes. Combining the Le2i and Fall Detection datasets allows for broader coverage of diverse postures and types of falls.

To further validate our model, we created an additional dataset with 24 videos of falls and normal activities (ADL) (Fig. 1c). This dataset includes four individuals of different ages and body sizes, recorded in four distinct locations under varying lighting conditions.

C. Dataset Preparation

We first processed and filtered the dataset by removing incorrect, incomplete, irrelevant, and duplicate samples to ensure effective model training. From the first dataset, we selected 2202 images and 27 videos, while from the second, we extracted 11930 images. In total, we gathered, from the three datasets, 14132 images and 51 videos for training, validation, and testing. However, the dataset was imbalanced, with only 3793 fall images compared to 11239 non-fall images. To prevent model bias during training, we applied data augmentation to the non-fall class. The augmentation process included the following transformations:

1. Horizontal flipping,
2. Image cropping, with a minimum zoom of 0% and a maximum zoom of 3%,
3. Saturation adjustment, ranging from -25% to +25%.

These pre-processing steps ensured a more balanced dataset and improved the robustness of the model in detecting falls under various conditions.

Table 1. Number of Fall Images before and After Data Augmentation

Augmentation Status	Fall images	Non-fall images	Total
Before data augmentation	3793	11239	14132
After data augmentation	7287	11239	18526

After applying data augmentation, the number of fall images increased to 7287, resulting in a more balanced dataset (Table 1). The final dataset consisted of 18526 images and 51 videos for training, validation, and test. The data were divided into three subsets as follows:

(A) Training Set (Train_Set)

- Comprises approximately 70% of the total dataset (12942 images)
- Used for training the model

(B) Validation Set (Validation_Set)

- Represents 20% of the dataset (3720 images)
- Used for model validation and hyperparameter tuning

(C) Test Set (Test_Set)

- Accounts for 10% of the dataset (1864 images and 51 videos)
- Used to evaluate the model's performance after training and validation

It should be noted that the images allocated for validation and testing were not subjected to data augmentation, ensuring an unbiased evaluation of the model's real-world performance.

Figure 3. Positions used to identify the subject posture. The situation in (a) represents a bounding box surrounding a standing person. The box can be in the middle, left or right of the image. The box can cover only part of the person. (b) shows a situation in which an object occludes the subject, so only the top of the body is detected and surrounded. In (c), bounding box surrounds a person in a lying position but not on the ground (on a bed, for example). In this situation, the classification model considers that the subject is not in a fall situation. (d) and (e) represent, respectively, fall states in which a person is horizontal and vertical positions.

For each image, the annotation process involved drawing bounding boxes around individuals whose postures needed to be recognized, as well as relevant parts of the surrounding environment. This step is crucial to the effectiveness of the proposed solution. Each bounding box was assigned a class label: "fall" or "non-fall." Special attention was given to selecting only the environmental features that could assist the model in accurately predicting fall situations. The annotation process was time-consuming, requiring multiple adjustments of bounding boxes and model retraining.

Fig. 3 shows different situations that the classification system might encounter and that we considered during the annotation process. Green boxes represent images, while blue boxes indicate those used for annotating the person and their immediate environment. To give the future model the ability to recognize occlusion situations, we were careful to include them. For example, in Fig. 3(a), we included images where only half of a person is visible. We repeated this process for the other situations. This improved the model's ability to recognize occlusion scenarios.

D. The Fine-tuned YOLOv10

The pre-trained deep learning model, YOLOv10 (You Only Look Once) [30], has been investigated in our study to detect and classify fall and non-fall situations in images and videos. YOLOv10 is a pioneering network designed for real-time object detection, known for its speed and accuracy. Unlike traditional methods that apply object recognition and localization in multiple stages, YOLOv10 processes the input image and predicts bounding boxes and class

probabilities directly from full images in a single network evaluation. This approach enables YOLOv10 to achieve high inference speeds while maintaining competitive accuracy, making it an ideal choice for real-time applications such as fall detection. YOLOv10 employs a unified detection framework that partitions the input image into multiple regions, each responsible for predicting bounding boxes along with corresponding class probabilities. The network leverages deep convolutional layers to extract rich spatial features and utilizes anchor boxes to effectively detect objects across varying scales and aspect ratios. YOLOv10 delivers excellent results in object detection and localization. By using a pre-trained YOLOv10 model trained on large-scale datasets (e.g., MS COCO), it retains the knowledge that it has already learned. Through fine-tuning YOLOv10, we teach it to perform fall/non-fall detection, demonstrating its effectiveness as a versatile and efficient tool for transfer learning. This fine-tuning exemplifies how YOLOv10's powerful capabilities can be leveraged to significantly improve the performance of object detection systems in new fields. The architecture of the proposed fine-tuned YOLOv10 model is illustrated in Table 2. The first step was adapting the model to the specific requirements of our dataset, which consists of RGB images of size 320×240. To do this, we changed the input shape of the model to match our dataset's dimensions and adjusted the number of output classes to two: "fall" and "non-fall." Additionally, we fine-tuned the anchor boxes to better fit the size and aspect ratios of the objects in our dataset. To avoid losing the valuable knowledge from the pre-trained YOLOv10 model while tailoring it for our fall detection task, we froze the initial layers and fine-tuned the later layers. To enhance the model's classification ability, we added custom layers to the head of the network (at the end of YOLOv10). Importantly, the core backbone of YOLOv10 remains unchanged. We froze the early backbone layers to preserve pre-trained features, retrained the later neck and head layers, and appended task-specific classification components including global average pooling followed by dense layers, dropout, and attention mechanisms. These additions enable robust contextual analysis of detected person regions while maintaining real-time performance.

Table 2. Summary of the Fine-tuned YOLOv10 Structure

Model Part	Layers	Frozen/Trained	Role
Input	Input Layer	-	Receives raw images (320x240x3)
Backbone	Conv + CSPDarknet (layers 0-9)	Frozen	Extract general features (edges, textures, shapes, patterns)
Neck	SPPF + PANet (first layers)	Frozen	Multi-scale feature aggregation using spatial pyramid pooling and initial feature fusion
Neck	PANet (last layers)	Trained	Adapt spatial features for fall detection
Head	Detection Heads (P3, P4, P5) + Regression layers	Trained	Detect objects at various scales and predict bounding boxes for person localization
Custom Layers	Global Average Pooling (GAP)	New/Trained	Dimensionality reduction to facilitate classification
	Dense Layer (256 units, ReLU)	New/Trained	Extract additional features for classification
	Dropout (0.3)	New/Trained	Reduce overfitting by randomly deactivating neurons
	Dense Layer (128 units, ReLU)	New/Trained	Further feature refinement for fall detection
	Dropout (0.3)	New/Trained	Additional regularization to prevent overfitting
	PSA (Partial Self-Attention)	New/Trained	Improves focus on critical regions (falling people)
	Sigmoid	New/Trained	Converts logits into probabilities for binary classification

These custom layers include additional convolutional layers for improved spatial feature extraction and a final detection layer for predicting bounding boxes and class probabilities. For training, we applied the Adam optimizer with a learning rate of 0.001, binary cross-entropy as the loss function, and mean squared error (MSE) loss for bounding box regression. The collective design choices resulted in a customized architecture ready for fall detection, utilizing the learned parameters from the pre-trained YOLOv10 layers along with the capabilities of the custom layers.

E. The FD-CNN Model

Convolutional Neural Networks (CNNs) are a type of deep neural network that can detect and classify falls within images. They are widely used for image classification and object detection. These networks automatically learn sophisticated patterns in the data by stacking trainable small filters, or kernels, on top of each other. This property reduces the need for expensive manual feature extraction methods. A typical CNN architecture consists of multiple convolutional layers, followed by various components such as batch normalization, non-linear activation functions, dropout layers, pooling layers, and a classification layer (usually a fully connected layer). This arrangement enables the network to extract features from the data in a hierarchical manner. CNNs offer remarkable performance while maintaining efficient computational processing, achieved through shared weight architectures and parallelization techniques [33].

Table 3. Summary of the Proposed FD-CNN Structure

Layer	Type	Parameters	Role
Input Layer	Input	320x240x3	Receive input images
Conv2D + ReLU	Convolution	32 filters, 3x3 kernel, ReLU activation	Extract low-level features (edges, textures)
MaxPooling2D	Pooling	Pool size 2x2	Reduce spatial dimensions
Conv2D + ReLU	Convolution	64 filters, 3x3 kernel, ReLU activation	Extract intermediate-level features
MaxPooling2D	Pooling	Pool size 2x2	Reduce spatial dimensions
Conv2D + ReLU	Convolution	128 filters, 3x3 kernel, ReLU activation	Extract high-level features
MaxPooling2D	Pooling	Pool size 2x2	Reduce spatial dimensions
Conv2D + ReLU	Convolution	256 filters, 3x3 kernel, ReLU activation	Extract complex features
Global Average Pooling (GAP)	Pooling	Global pooling	Reduce spatial dimensions to a feature vector
Dense + ReLU	Fully Connected	128 units, ReLU activation	Transform features into a vector for classification
Dropout	Regularization	Rate of 0.5	Reduce overfitting
Sigmoid	Fully Connected	1 unit, sigmoid activation	Produce a probability for the "fall" class

As shown in Table 3, we propose a custom Fall Detection CNN (FD-CNN) model, which contains layers specifically helpful for feature extraction and fall/non-fall image classification. The architecture starts with an input layer that receives an RGB image of size 320×240 pixels with 3 channels (representing the 3 colour values: red, green, and blue). The input is then passed through four convolutional layers, each followed by max-pooling layers. The first convolutional layer has 32 filters with a kernel size of 3×3 and a stride of 1, accompanied by a ReLU activation function. To downsample the feature map, a max-pooling layer with a pool size of 2×2 and a stride of 2 is applied. The second convolutional layer has 64 filters, each of size 3×3 with a stride of 1, followed by a ReLU activation function. This is again followed by a max-pooling layer with a 2×2 pool size and stride of 2 for further downsampling. The third

convolutional layer contains 128 filters with a kernel size of 3×3 and a stride of 1, followed by a ReLU activation function. A max-pooling layer with a 2×2 pool size and a stride of 2 is applied again. The fourth convolutional layer consists of 256 filters with a kernel size of 3×3 and a stride of 1, followed by a Global Average Pooling (GAP) layer to summarize the spatial features into a compact feature vector. The flattened output from these convolutional and pooling layers is fed into a dense layer comprising 128 neurons with ReLU activation and a dropout layer with a rate of 0.5 to prevent overfitting. Finally, the output layer uses sigmoid activation for binary classification (fall/non-fall). For training, the model uses the Adam optimizer with a learning rate of 0.001 and a binary cross-entropy loss function to measure the difference between predicted and actual class probabilities.

4. Experiments and Evaluations

This section presents the experimental results of using the proposed deep learning models to detect and classify falls. To ensure the robustness and reliability of our results, the training of the CNN and YOLOv10 models was repeated five times with different dataset distributions for each run. This method helped us understand variations in the training process and ensured the stability and generalizability of the models. The performance of the models was subsequently evaluated, and average metrics were calculated for all runs. Using average performance metrics helps prevent biases caused by a single training run. Thus, saving the weights that gave the highest validation accuracy in each repetition serves as an important safeguard against potential model overfitting and ensures that we retain the most optimal models for evaluation. We used checkpoints callback throughout these repetitions to save the best state of the models during training. Realizing the importance of fall detection, we focused on achieving the highest accuracy in classifying fall and non-fall situations with the single best model, based on the highest accuracy on the validation set. We used early stopping callbacks based on validation loss and accuracy, which allowed us to terminate the training process if no improvement in validation metrics was observed over a specified number of epochs. This prevented overfitting and ensured better generalization of the model.

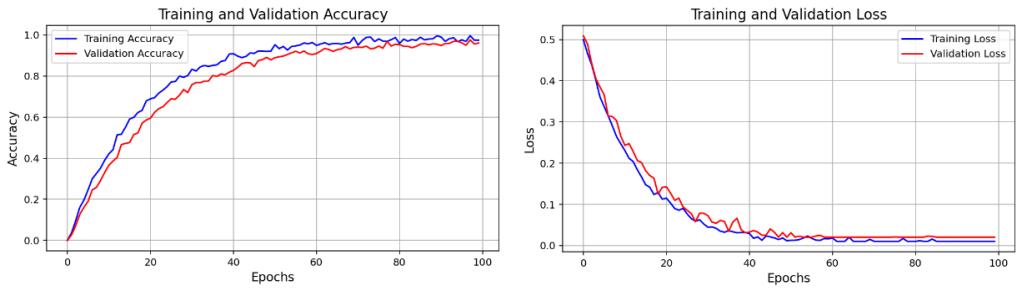


Figure 4. Train and validation accuracy/loss of YOLOv10

Table 4. The YOLOv10 Model Evaluation Performance Metrics

Test	Accuracy	Fall			Not Fall		
		Precision	Recall	F1 score	Precision	Recall	F1 score
1	98.93%	97.98%	99.32%	98.65%	99.55%	98.67%	99.11%
2	**99.03%**	**97.99%**	**99.59%**	**98.78%**	**99.73%**	**98.67%**	**99.20%**
3	98.93%	97.85%	99.45%	98.64%	99.64%	98.59%	99.11%
4	98.87%	97.85%	99.32%	98.58%	99.55%	98.58%	99.06%
5	98.82%	97.85%	99.18%	98.51%	99.46%	98.58%	99.02%
Average	98.92%	97.90%	99.37%	98.63%	99.59%	98.62%	99.10%

In the context of our proposed models, we started by testing YOLOv10 for fall detection. YOLOv10 showed exceptional performance with a training accuracy of 99.5% and a validation accuracy of 99.21% (Fig. 4). Although it achieved high accuracy, it required substantial

training time to reach peak performance (100 epochs, averaging 4 hours on Google Colab). The loss function consistently decreased during training before stabilizing (Fig. 4).

Table 5. The YOLOv10 Model Evaluation Performance Metrics on Videos

Test	Correct detection		Incorrect detection	
	Fall videos	ADL videos	Fall videos	ADL videos
1	22	26	3	0
2	23	26	2	0
3	22	26	3	0
4	22	26	3	0
5	22	26	3	0

Next, we implemented and tested the FD-CNN. After 50 epochs of training, the CNN showed a training accuracy of 95.01% and a validation accuracy of 94.92% (Fig. 5). The loss function kept decreasing during training and then stabilized (Fig. 5). The FD-CNN model required less training time to reach this accuracy (an average of 2 hours on Google Colab). This FD-CNN architecture was the best configuration that gave us the best results. In the evaluation, both YOLOv10 and FD-CNN models performed very well in detecting fall and non-fall situations with good accuracy, precision, recall, and F1 scores. However, YOLOv10 had a noticeable advantage; it showed a high level of consistency with a high accuracy level in classifying fall and non-fall in different training repetitions. We can see in Table 4 that accuracy remained consistently high, within the range of 98.82% - 99.03%, showing the remarkable efficiency of the model with an average accuracy of 98.92%. The precision, recall, and F1 scores of both YOLOv10 and FD-CNN models demonstrated the ability to separate fall and non-fall situations very accurately.

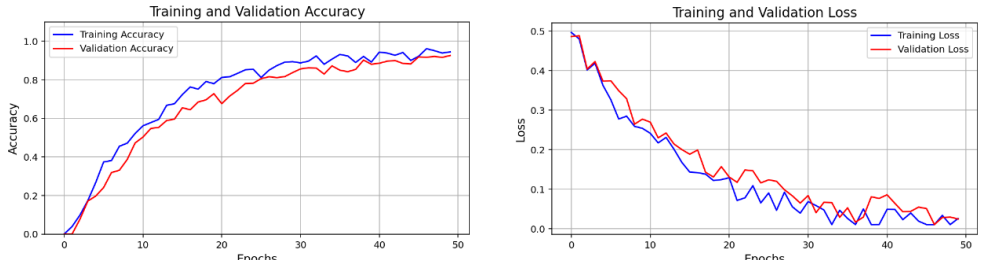


Figure 5. Training and Validation accuracy/loss of FD-CNN.

In contrast, the FD-CNN model trained from scratch, showed lower performance with accuracy between 93.94% and 94.10%. Precision, recall, and F1 scores were also lower for both fall and non-fall classes. The FD-CNN model performed less effectively with fall events, as demonstrated by its lower precision, recall, and other metrics (Table 6). This discrepancy is likely due to the relatively small dataset size, which may not be enough for the FD-CNN to generalize well compared to the pre-trained YOLOv10. Even though their performances were not the same, both models were very stable across different runs, with negligible changes in metrics. YOLOv10 outperformed FD-CNN in classifying fall and non-fall events, consistently achieving high precision, recall, and F1 scores, making it a highly reliable choice for real-time fall detection applications. FD-CNN showed some potential, but there is a larger performance gap, especially in detecting fall situations, which suggests that using a bigger dataset would greatly improve its performance. Tables 4 and 6 summarize the average performance metrics for YOLOv10 and FD-CNN, respectively, including accuracy, precision, recall, and F1 score. Tables 5 and 7 show tests on the 51 videos from the test set. Additionally, Fig. 6 presents the confusion matrices for the best evaluation of both models, offering further insight into their classification outcomes. The consistently high precision, recall, and F1 score for YOLOv10

suggest that it is the more dependable and accurate choice for fall detection compared to the FD-CNN model.

Table 6. The CNN model evaluation performance metrics

Test	Accuracy	Fall			Not Fall		
		Precision	Recall	F1 score	Precision	Recall	F1 score
1	94.05%	91.51%	93.17%	92.33%	95.68%	94.59%	95.13%
2	93.99%	91.51%	93.04%	92.27%	95.59%	94.59%	95.09%
3	**94.10%**	**91.63%**	**93.17%**	**92.39%**	**95.68%**	**94.68%**	**95.18%**
4	94.05%	91.63%	93.04%	92.33%	95.59%	94.68%	95.13%
5	93.94%	91.41%	93.06%	92.23%	94.49%	95.58%	95.03%
Average	94.02%	91.55%	93.08%	92.30%	95.34%	94.88%	95.11%

Table 7. The CNN Model Evaluation Performance on Videos

Test	Correct detection		Incorrect detection	
	Fall videos	ADL videos	Fall videos	ADL videos
1	21	23	5	2
2	20	23	5	3
3	21	24	4	2
4	20	23	5	3
5	21	24	4	2

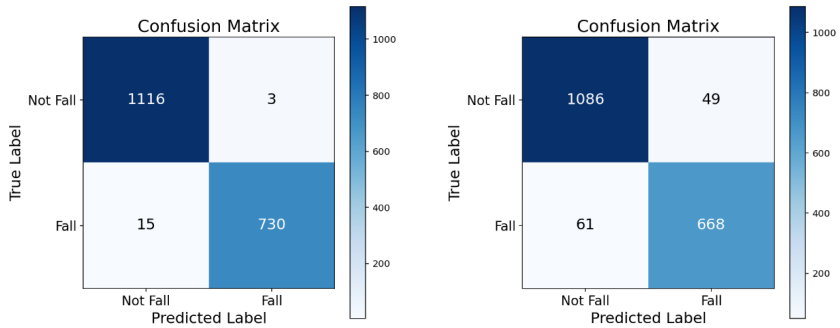
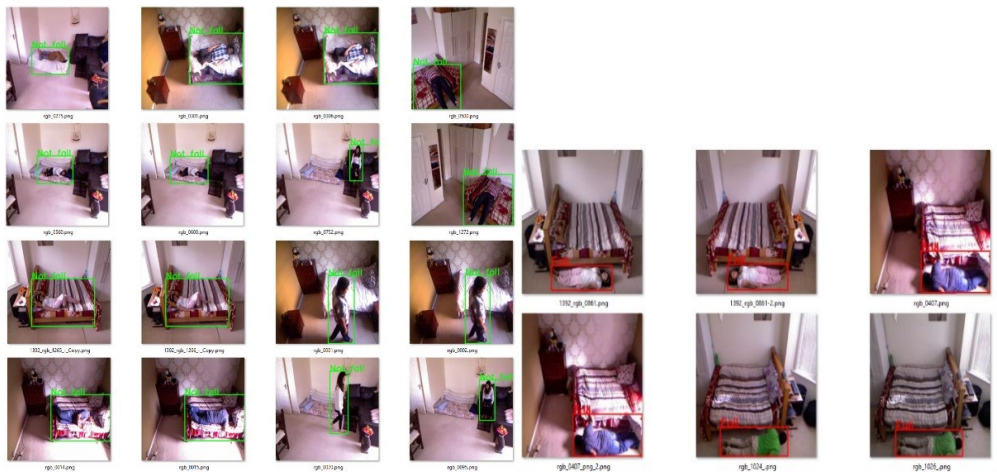


Figure 6. Confusion matrix of YOLOv10 and CNN on Test_set_images.



(a) (b)

Figure 7. Results on Test_set

5. Ablation study

In this section, we conduct an ablation study to analyse the impact of different components within our proposed approach on the achieved results. Specifically, we examine how considering the person's surroundings and occlusions during annotation process affects the performance of both YOLOv10 and the FD-CNN models. We assess three different configurations: excluding both the person's surroundings and occlusions, excluding only the person's surroundings, and excluding only occlusions, comparing them to the full proposed approach. As presented in Table 8, the results show that excluding the person's surroundings during annotation leads to a decrease in accuracy, precision, recall, and F1-score for both models. This highlights the importance of incorporating contextual information to enhance classification performance. However, despite this decline, the performance of both models remains relatively high. YOLOv10 maintains an accuracy exceeding 95.76%, demonstrating its robustness even without additional contextual information.

Table 8. Ablation study of our proposed approach, assessing the impact of accounting for a person's surroundings and occlusions during the annotation process.

Method	Acc.	Fall			Not Fall		
		Prec.	Rec.	F1	Prec.	Rec.	F1
YOLOv10 without consideration of the person's surroundings	95.76	91.01	97.92	94.34	98.77	94.55	96.61
YOLOv10 without consideration of occlusions	96.89	93.78	98.12	95.90	98.86	96.16	97.49
YOLOv10 without consideration of person's surroundings and occlusions	94.53	88.24	97.40	92.59	98.51	92.97	95.66
CNN without consideration of the person's surroundings	91.42	84.77	90.73	87.65	95.68	88.54	91.97
CNN without consideration of occlusions	92.33	87.11	92.84	89.88	95.68	92.03	93.82
CNN without consideration of person's surroundings and occlusions	90.88	84.77	91.29	87.91	94.80	90.65	92.68
Proposed full YOLOv10	98.92	97.90	99.37	98.63	99.59	98.62	99.10
Proposed full CNN	94.02	91.55	93.08	92.30	95.34	94.88	95.11

In contrast, the FD-CNN model experiences a more noticeable drop in accuracy and recall, indicating a greater sensitivity to variations in the dataset and a stronger dependence on contextual cues for accurate classification. A similar pattern is observed when considering occlusions. The models achieve slightly lower precision and recall, particularly for fall detection, as occluded instances present additional challenges in distinguishing falls from non-falls. However, the F1-score remains relatively stable for YOLOv10, reinforcing its strong generalization capabilities. The FD-CNN model, on the other hand, exhibits a greater fluctuation in F1-score, suggesting a higher dependency on visual cues. Furthermore, when both the person's surroundings and occlusions are ignored during annotation, the performance degradation becomes even more pronounced. In this case, accuracy, precision, recall, and F1-score further decrease for both models. Nevertheless, across all these scenarios, including full annotation, exclusion of surroundings, exclusion of occlusions, and exclusion of both surroundings and occlusions, YOLOv10 consistently outperforms the CNN model. This confirms its superior ability to generalize and maintain high classification performance, even when critical contextual information is omitted. Overall, this ablation study highlights the importance of our proposed annotation strategy. While considering the person's surroundings and occlusions introduces additional complexity, the full approach achieves the best balance across accuracy, precision, recall, and F1-score. These findings confirm that our proposed

method enhances the models' ability to handle challenging scenarios, making them more effective for real-world fall detection applications.

6. Performance Comparison

The comparative analysis in Table 9 highlights the key differences between our proposed approach and existing solutions for fall detection. As shown in the second column, prior methods fall into two main categories: vision-based and wearable-based in addition to ambient-based. Approaches using wearable sensors, such as accelerometers and gyroscopes [11] and [18], achieve high F1-scores exceeding 97%. Since they are attached to the person, the occlusion problem is eliminated. However, they require users to wear a device, which may be inconvenient. Moreover, these methods lack context awareness and cannot distinguish between an actual fall and other similar movements. Ambient sensor-based methods, such as microphones [29] and radars [28], eliminate the need for wearable devices, but they are highly sensitive to ambient noise, interference and occlusions. Additionally, they struggle to accurately analyse a person's posture, leading to increased false positives and false negatives.

The most effective solutions leverage RGB cameras combined with deep learning algorithms such as [20][22][24][25][26][27][35] and [36]. Some approaches achieve outstanding results, such as [25] with an Accuracy of 99.93% and a F1-score of 99.92%, but they suffer from major limitations:

1. They do not handle occlusions, meaning a fall may go undetected if the person is partially obscured by an object or furniture.
2. They cannot differentiate between a fall and a person lying on a bed or mattress, leading to a high rate of false positives, except for the research in [10], which used a simpler solution based on the person's height in the image. However, this solution can be easily biased by the person's distance from the camera, making it an unreliable approach to this problem.

Our approach, based on YOLOv10 and CNN, effectively addresses these challenges. Unlike other methods, it reliably detects falls even in the presence of occlusions and accurately distinguishes between a person lying on the floor and someone resting on a bed or mattress, significantly reducing false alarms. With an accuracy of 98.92% and an F1-score of 98.78%, our solution ensures superior robustness and reliability in real-world scenarios. By incorporating these advanced capabilities while maintaining top-tier performance, our approach represents a major breakthrough in automated fall detection, ensuring better safety for vulnerable individuals and minimizing unnecessary emergency responses.

Table 9. Comparison of our Work with the State of the Art

Ref.	Input signal	Dataset	Occlusion	Detection of rest on bed, ...	Technique	Performance
[11]	IMU	MobiAct, Sisfall	NOP	No	GCN + CNN	Accuracy: 99.5% F1-score: 98.9%
[13]	Wearable	KFall, SisFall	NOP	NOP	CNN	Recall: 99%
[18]	Accelerometer	Local	NOP	No	LSTM, Transformer encoders	Accuracy: 98% F1-score: 97%
[34]	Accelerometer gyroscope (Wrist)	UP-FALL, WEDA Fall, UMA Fall	NOP	No	ANN, SVM	Recall: 90.57% F1-score: 69.93%
[28]	Radar	Local	No	No	MLP, CNN, RF, KNN, SVM	Accuracy: 92.3% Recall: 89.2% Precision: 80.01% F1-score: 84.4%
[29]	Microphone	Not	NOP	No	Hidden	Accuracy: 97%

Ref.	Input signal	Dataset	Occlusion	Detection of rest on bed, ...	Technique	Performance
		mentioned			Markov Model	
[10]	RGB camera	IASLAB-RGBD, URFD	No	Yes	Proposed algorithm	Precision: 99.01%
[20]	RGB camera	URFD, local	No	No	STGCN and 1D-CNN	Accuracy: 96.53%
[22]	RGB camera	UP-Fall, ADLF	No	No	CNN, GRU, ANN	Accuracy: 96.7%
[24]	RGB camera	Le2iFD, URFD	No	No	Temporal Convolution Network, SVM, LSTM	Accuracy: 97.02%
[25]	RGB camera	UP-Fall	No	No	CNN	Accuracy: 99.93%
[26]	RGB camera	UR Fall, FDD, Multiple Cameras Fall Dataset	No	No	Transfer learning on YOLOv5	Precision: 94% Recall: 88%
[27]	RGB camera	UP-FALL	No	No	RF, SVM, MLP, KNN, AlphaPose	Accuracy: 99.34%
[35]	RGB camera	UR Fall, FDD	No	No	LSTM, Gated Recurrent Unit	Accuracy: 98.2%
[36]	RGB camera	Not necessary	No	No	YOLOv7-W6-Pose, proposed algorithm	Accuracy: 96.15%
[37]	RGB camera	Not necessary	No	No	OpenPose, proposed algorithm	Accuracy: 97%
[38]	RGB camera	MultiCam fall, Le2i, MS COCO Keypoints	Yes	No	ANN	Precision: 90.08%
Proposed	RGB camera	Le2i, Fall detection dataset	Yes	Yes	YOLOv10, CNN	Accuracy: 98.92%

*NOP: No Occlusion Problem

7. Conclusion

This paper introduces a real-time, vision-based fall detection system using a single RGB camera, effectively addressing critical challenges such as partial occlusions and the common false positives arising from distinguishing actual falls on the floor from non-fall scenarios like resting on a bed, sofa, or similar furniture. By leveraging transfer learning on a fine-tuned YOLOv10 model and a custom CNN architecture, combined with multi-dataset training (resulting in 18526 pre-processed images and 51 videos), the system achieves robust contextual awareness and high detection accuracy. Evaluation on 1864 test images and the full video set demonstrates superior performance, with the YOLOv10-based approach attaining an outstanding overall accuracy of 98.92%. These results underscore the efficacy of our one-step, context-aware method as a reliable, scalable solution for elderly care applications.

Although the proposed system exhibits strong performance across diverse scenarios, certain limitations remain. Highly unusual room configurations or environments lacking conventional furniture can reduce contextual discrimination due to insufficient or unfamiliar environmental cues in the training data. Similarly, some situations in which the person does not exhibit a clear horizontal collapse pose remain challenging for analysis. Finally, full occlusion (e.g., a person completely hidden under furniture) represents a fundamental limitation of monocular RGB-based systems.

Future research will directly target these limitations to enhance robustness and practical deployment. For instance, integrating multi-camera setups or advanced temporal modelling could better handle vertical falls and unusual configurations. To overcome full occlusions and improve generalization, incorporating multimodal sensors, such as depth cameras or radar, offers promising avenues for richer environmental understanding and reduced false positives. Additionally, deploying optimized models on edge devices for low-latency, privacy-preserving inference in IoT-enabled healthcare systems remains a key direction for real-world adoption.

8. References

[1]. WHO, "Falls," World Health Organization, Apr. 26, 2021. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/falls . [Accessed: Jan. 2, 2025].

[2]. The National Council on Aging, "National Council on Aging." [Online]. Available: https://www.ncoa.org/ . [Accessed: Dec. 21, 2024].

[3]. X. Wang, J. Ellul, and G. Azzopardi, "Elderly fall detection systems: a literature survey," Front. Robot. AI, vol. 7, p. 71, Jun. 2020, doi: 10.3389/frobt.2020.00071.

[4]. Md. M. Islam et al., "Deep learning based systems developed for fall detection: a review," *IEEE Access*, vol. 8, pp. 166117–166137, 2020, doi: 10.1109/ACCESS.2020.3021943.

[5]. N. T. Newaz and E. Hanada, "The methods of fall detection: a literature review," Sensors, vol. 23, no. 11, p. 5212, May 2023, doi: 10.3390/s23115212.

[6]. A. Ramachandran and A. Karuppiah, "A survey on recent advances in wearable fall detection systems," BioMed Research International, vol. 2020, no. 1, p. 2167160, Jan. 2020, doi: 10.1155/2020/2167160.

[7]. S. Usmani, A. Saboor, M. Haris, M. A. Khan, and H. Park, "Latest research trends in fall detection and prevention using machine learning: a systematic review," Sensors, vol. 21, no. 15, p. 5134, Jul. 2021, doi: 10.3390/s21155134.

[8]. N. Thakur and C. Y. Han, "A study of fall detection in assisted living: identifying and improving the optimal machine learning method," JSAN, vol. 10, no. 3, p. 39, Jun. 2021, doi: 10.3390/jsan10030039.

[9]. J. Gutiérrez, V. Rodríguez, and S. Martin, "Comprehensive review of vision-based fall detection systems," Sensors, vol. 21, no. 3, p. 947, Feb. 2021, doi: 10.3390/s21030947.

[10]. S. Lafuente-Arroyo et al., "RGB camera-based fallen person detection system embedded on a mobile platform," *Expert Systems with Applications*, vol. 197, p. 116715, Jul. 2022, doi: 10.1016/j.eswa.2022.116715.

[11]. Y. Li, Z. Zuo, and J. Pan, "Sensor-based fall detection using a combination model of a temporal convolutional network and a gated recurrent unit," Future Generation Computer Systems, vol. 139, pp. 53–63, 2023, doi: 10.1016/j.future.2022.09.011.

[12]. S. Wang et al., "Human motion recognition by a shoes-floor triboelectric nanogenerator and its application in fall detection," Nano Energy, vol. 108, p. 108230, 2023, doi: 10.1016/j.nanoen.2023.108230.

[13]. X. Yu et al., "A practical wearable fall detection system based on tiny convolutional neural networks," *Biomedical Signal Processing and Control*, vol. 86, p. 105325, 2023, doi: 10.1016/j.bspc.2023.105325.

[14]. M. Musci et al., "Online fall detection using recurrent neural networks on smart wearable devices," *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 3, pp. 1276–1289, Jul.–Sep. 2021, doi: 10.1109/TETC.2020.3027454.

[15]. S. Nooruddin, M. Islam, and F. Sharna, "An IoT based device-type invariant fall detection system," Internet of Things, vol. 9, p. 100130, 2020, doi: 10.1016/j.iot.2019.100130.
[16]. A. H. Ngu et al., "Personalized fall detection system," in 2020 *IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), Austin, TX*, USA, 2020, pp. 1–7, doi: 10.1109/PerComWorkshops48775.2020.9156172.
[17]. M.-H. Chang et al., "Cross-platform gait analysis and fall detection wearable device," Applied Sciences, vol. 13, no. 5, p. 3299, Mar. 2023, doi: 10.3390/app13053299.
[18]. H. Yhdego, C. Paolini, and M. Audette, "Toward real-time, robust wearable sensor fall detection using deep learning methods: a feasibility study," *Applied Sciences*, vol. 13, no. 8, p. 4988, Apr. 2023, doi: 10.3390/app13084988.
[19]. K. Fei et al., "Flow-pose Net: An effective two-stream network for fall detection," The Visual Computer, vol. 39, no. 6, pp. 2305–2320, 2023, doi: 10.1007/s00371-022-02416-2.
[20]. M. Amsaprabhaa et al., "Multimodal spatiotemporal skeletal kinematic gait feature fusion for vision-based fall detection," *Expert Systems with Applications*, vol. 212, p. 118681, 2023, doi: 10.1016/j.eswa.2022.118681.
[21]. A. Patel et al., "AI-powered trustable and explainable fall detection system using transfer learning," Image and Vision Computing, vol. 149, p. 105164, 2024, doi: 10.1016/j.imavis.2024.105164.
[22]. S. Yadav et al., "ARFDNet: An efficient activity recognition & fall detection system using latent feature pooling," Knowledge-based Systems, vol. 239, p. 107948, 2022, doi: 10.1016/j.knosys.2021.107948.
[23]. Y. Galvao et al., "A multimodal approach using deep learning for fall detection," Expert Systems with Applications, vol. 168, p. 114226, 2021, doi: 10.1016/j.eswa.2020.114226.
[24]. D. R. Beddiar, M. Oussalah, and B. Nini, "Fall detection using body geometry and human pose estimation in video sequences," *Journal of Visual Communication and Image Representation*, vol. 82, p. 103407, Jan. 2022, doi: 10.1016/j.jvcir.2021.103407.
[25]. P. Qi, D. Chiaro, and F. Piccialli, "FL-FD: Federated learning-based fall detection with multimodal data fusion," *Information Fusion*, vol. 99, p. 101890, Nov. 2023, doi: 10.1016/j.inffus.2023.101890.
[26]. Z. Luo et al., "Elderly fall detection algorithm based on improved YOLOv5s," ITC, vol. 53, no. 2, pp. 601–618, Jun. 2024, doi: 10.5755/j01.itc.53.2.36336.
[27]. H. Ramirez et al., "Fall detection and activity recognition using human skeleton features," *IEEE Access*, vol. 9, pp. 33532–33542, 2021, doi: 10.1109/ACCESS.2021.3061626.
[28]. A. Rezaei et al., "Unobtrusive human fall detection system using mmwave radar and data driven methods," *IEEE Sensors Journal*, vol. 23, no. 7, pp. 7968–7976, Apr. 2023, doi: 10.1109/JSEN.2023.3245063.
[29]. J. Lian, X. Yuan, M. Li, and N.-F. Tzeng, "Fall detection via inaudible acoustic sensing," Proc. ACM Interact. Mob. Wearable Ubiquitous Technol., vol. 5, no. 3, pp. 1–21, Sep. 2021, doi: 10.1145/3478094.
[30]. A. Wang et al., "YOLOv10: Real-time end-to-end object detection," Advances in Neural Information Processing Systems, vol. 37, pp. 107984–108011, 2025.
[31]. "Le2i Fall Dataset," [Online]. Available: https://www.kaggle.com/datasets/tuyenldvn/falldataset-imvia . [Accessed: Nov. 21, 2024].
[32]. "Fall detection Dataset," [Online]. Available: https://falldataset.com/ . [Accessed: Nov. 18, 2024].
[33]. S. Hong et al., "Opportunities and challenges of deep learning methods for electrocardiogram data: A systematic review," Comput. Biol. Med., vol. 122, p. 103801, Jul. 2020, doi: 10.1016/j.compbiomed.2020.103801.
[34]. V. Fula and P. Moreno, "Wrist-based fall detection: towards generalization across datasets," Sensors, vol. 24, no. 5, p. 1679, 2024, doi: 10.3390/s24051679.

[35]. C.-B. Lin et al., “A framework for fall detection based on openpose skeleton and LSTM/GRU models,” *Applied Sciences*, vol. 11, no. 1, p. 329, Dec. 2020, doi: 10.3390/app11010329.
[36]. E. Tîrziu et al., “Enhanced fall detection using YOLOv7-W6-Pose for real-time elderly monitoring,” Future Internet, vol. 16, no. 12, p. 472, Dec. 2024, doi: 10.3390/fi16120472.
[37]. W. Chen, Z. Jiang, H. Guo, and X. Ni, “fall detection based on key points of human-skeleton using openpose,” Symmetry, vol. 12, no. 5, p. 744, May 2020, doi: 10.3390/sym12050744.
[38]. A. Umar, S. Von Cavallar, J. Tang, and S. Harrer, “SSHFD: single shot human fall detection with occluded joints resilience,” Frontiers in Artificial Intelligence and Applications, IOS Press, 2020, doi: 10.3233/FAIA200403.

Fouzi Lezzar is currently an associate professor in the department of Software Technologies and Information Systems, Faculty of New Technologies of Information and Communication, University Constantine 2, Algeria. He holds a PhD in Computer science from Batna 2 University (Batna - Algeria). His research interest includes Artificial intelligence, Internet of Things, smart cities and e-health.

Seif Eddine Mili is an associate professor in the Department of Computer Science, Ecole Normale supérieure Constantine, Algeria. He holds a PhD in Computer science from Badji Mokhtar Annaba University, Algeria. His research interest includes Artificial intelligence, Web of Things, IoT, Multi Agent Systems; Bio-inspired Systems and Model Driven Architecture.

Nabila Azeri is a researcher in the field of computer science, affiliated with Abbes Laghrour University, Algeria, where she earned her PhD in Computer Science. Her research focuses on cyber-physical systems (CPS), artificial intelligence, and privacy preserving technologies. She has made significant contributions to areas such as self-adaptive cyber-physical systems, federated learning, fault prediction, and enhancing resilience in dynamic systems.

Djamel Benmerzoug received the Ph.D. degree in computer science from Pierre and Marie Curie University, Paris, France. He is currently a Full Professor with the Department of TLSI, Faculty of New Technologies of Information and Communication, University Constantine 2, Algeria. He has published many papers in many international conferences and journals. He supervises many master’s and Ph.D. students. His current research interests include the Internet of Things, cloud computing, advanced enterprises systems, multi-agent systems, service-oriented computing, and business processes modelling and verification.