Final Course Project, CS4152, 2025.

Data of Submission 20/01/2025

Video Based Fall Detection System using YOLO

SYED BURHAN AHMED[1], ANOUSHA SHAKEEL[2], AND MAIRA RAUF.[3], (Members, UMT)

[1]Department of Artificial Intelligence, University of Management and Technology, Lahore, Pakistan)

"This work was supported by Dr Shaista Habib - Program Head AI, Department of Artificial Intelligence, University of Management and Technology under Course CS4152."

ABSTRACT To determine that individuals and more so mobility-impaired and elderly people, are secure and safe, fall detection systems have become really important. This paper introduces a high precision video based fall detection system that is using the capabilities of You Only Look Once(YOLO) model version 11 object detection model created by Ultralytics and further trained on the subset of LE2I Fall Detection Dataset. The model is trained for real time fall detection. The system has integrated advanced Computer Vision and Deep Learning techniques. This will help for the fall detection on videos, thus timely alerts are raised. The LE2I dataset , with its vast scenarios, was preprocessed and divided into test, train and validation subsets. The YOLO version 11 model was fine-tuned with 100 epochs, applying Adam optimizer and the learning rate of 0.001 and its performance was calculated using metrics like mean Average Precision(mAP) score, precision and recall. The trained model was tested on unseen videos, where it's prediction were visualized through annotated boxes. In addition, the results and model weights were archived for future use and deployment. The system is capable to detect falls in real time with high authenticity. We can integrate the systems into healthcare facilities, surveillance systems and smart homes to detect real time falls of specially small children and elderly enhancing safety.

INDEX TERMS Fall Detection, YOLO (You Only Look Once), HAR (Human Activity Recognition), Video Surveillance, Deep Learning, Computer Vision, Real-Time Detection, Action Recognition, Object Detection, Machine Learning, LE2I Dataset, Bounding Boxes, Feature Extraction, Image Processing, Neural Networks, Pre-trained Models, High Recall Systems, Safety Monitoring, Emergency Rooms (ER), World Health Organization (WHO), Centers for Disease Control and Prevention (CDC).

I. INTRODUCTION

FALLS are the leading cause of fatal injuries especially to elderly and create serious problems to those living independently [1].The World Health Organization(WHO) estimates that falls are the second biggest cause of unintended injuries leading to fatalities worldwide. Out of these, according to WHO, around **37.3 million falls** were severe enough that lead to medical interventions [2]. The Center for Disease Control and Prevention (CDC) reports that in the United States alone, one of four elderly experiences a fall yearly, that results in over **3 million ER visits** and around 50 billion dollars in medical bills [3].These statistics highlight the crucial need of effective real time fall detection systems, especially for elder population, to prevent fatal injuries and even deaths, reduce healthcare troubles and improve quality of life. Thus, fall detection is one of the most sought application for home monitoring and elderly care.

Without real-time fall detection systems it may happen that a minor fall would lead to a serious injury or long term immobility. Most of the earlier fall detection systems were based on wearable sensors or devices equipped with gyroscopes and accelerometers. However, all these systems pose very significant limitations including discomfort, non-compliance of users and lack of ability to monitor substantial environments. Moreover, lots of existing vision-based systems exhibit low accuracy in differentiating between falls and action similar to falls such as sitting or lying down, resulting in high false positive rates [4].

These challenges necessitate the need for **State-Of-The-Art and non-intrusive fall detection methodologies** that uses the latest technologies in order to deliver accurate, efficient and scalable solutions. A huge limitation exists in systems that can detect falls in real-time with minimalistic hardware requirements and high reliability across varying scenarios.

A. PROJECT SCOPE

The planned system uses **SOTA (State-Of-The-Art) deep learning techniques** to detect falls in videos frame by frame, ensuring high levels of accuracy and real-time performance. The system's crucial applications include smart homes specifically for elders and mobility restricted individuals, healthcare monitoring and public surveillance, providing a reliable and automated pathway to enhance safety and facilitate timely interventions.

Through this research, we aspire to fill the limitations in the existing solutions by presenting a practical, low cost and scalable fall detection model that is backed by real-world testing and experimentation.

B. OBJECTIVE

The primary objective of our project is to build a robust video-based fall detection system using **YOLO11**(You Only Look Once version 11) object detection model [5],to accurately depict fall in video streams in real-time to ensure timely intervention and enhancement of security for elderly and mobility constrained individuals and it is trained on the publicly available **LE2I Fall Detection Dataset.** [6]

This dataset contains a diverse collection of videos depicting fall and other activities such as walking, standing, sitting as non-fall activities, making it the ideal choice for training a State-Of-The-Art model.

C. SOLUTION

The initial idea of our research was to use a pre-trained model on the UCF-101 Dataset [7] as **fixed feature detector** and then further add some hidden layers of a neural network to perform classification of fall activities and other action activities like throwing disc, standing running as non-fall activities.

But after reviewing the options researching on our problem, we found out about the LE2I Dataset [6]. We thoroughly researched and then got to to know about **YOLO**(You Only Live Once) Algorithm and its performance and high accuracy scores, so we changed our idea to fine tuning and **YOLO**(You Only Live Once)'s latest version which was version 11 [5]. So first we researched and found out a model [9] already implemented and deployed using YOLO version 8 [8] on a subset of LE2I dataset [6] but it was not performing good as it was incorrectly classifying fall activities as non-fall and non-fall activities as fall.

So we used that dataset and modified it and then implemented YOLOv11(You Only Look Once version 11) object detection model's nano version named **yolo11n.pt**. Then we evaluated our model and tested it on unseen videos from the LE2I dataset [6] itself and some real-life videos we recorded ourselves and results it produced were very exceptional. On all the thresholds it was performing very exceptional except when the camera was moving too then more than one objects were detected but there was only one.

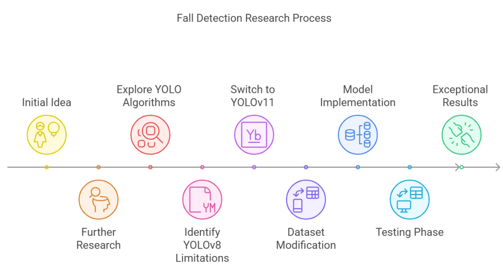

FIGURE 1: This figure shows the process workflow, explaining the stages from research to the implementation of YOLOv11 and testing on unseen and real-life video data.

II. LITERATURE REVIEW

Fall Detection has gained quite an attention particularly in the recent years, primarily for elderly care, surveillance systems and health monitoring. With aging population, robust fall detection systems are very important in providing on time responses in fatal situations to minimize the fatality by reducing the time for timely intervention.

These systems can drastically improve quality of life and reduce risk for grave injuries for minimal falls. These systems depend heavily on computer vision and neural networks to detect falls in real-time from video based data.

A. DATASETS FOR FALL DETECTION

There are many vision-based datasets available for fall detection systems like LE2I [7] which is widely available and features real-life scenarios from multiple angles with each frame annotated and it is very diverse as it includes difficult cases such as varied lighting. The vision-based approaches can base on any camera such as a web camera, a depth camera like Kinect and even a normal RGB camera. Many actions like changes of direction carrying an object and light on/off were simulated. Although publicly available at first, it is no where to be found.

One of the vision-based dataset is SDUFall [10] Dataset that uses a single Kinect camera to capture falls and five daily life activities performed by ten young men and women.

Zhang et al. [11] conferred 2 datasets that were collected with two points of view with two Kinect cameras. In the EDF dataset ten people did two falls in each point of view and for each of eight directions and also recorded more actions similar to falling like sitting down,laying, picking up something, etc. The OCCU dataset was focused to collect falls data again with two Kinect cameras. Five people performed around sixty falls that were occluded and also identical actions like the EDF dataset.

Charfi et al. [12] also conferred a dataset recorded with a single RGB camera containing normal activities, falls in four distinct locations. This dataset presented arrangements in about four distinct locations and falls in various angles.

Title	Authors	Methodology	Findings	Limitations
Falls Prediction Based on Body Keypoints and Seq2Seq Architecture [16]	Minjie Hua, Yibing Nan, and Shiguo Lian	This paper employs a sequence-to-sequence model with some LSTM layers to analyze and predict the key points of body joints. The encoder processes input sequences and decoder generate output sequences while both use sequence-to-sequence loss function.	The findings of this research show that classification model just using body key points outperforms many models based on basic raw RGB data. The inclusion of pose prediction enhances the system to predict a fall before it happens.	The performance of the model on datasets and real world scenarios other than Le2i dataset is not studied and the robustness of the system in varying lighting conditions and occlusions is not studied.
Human Fall Detection for Smart Home Caring Using YOLO Networks [17]	Bo Luo.	The method used in this research paper was using YOLO-based Networks particularly YOLOv5 and YOLOv6 for thei performance and accuracy and a custom dataset of 1425 was created and augmented using Roboflow and the models were trained and validated using transfer learning and leveraging pre-trained COCO weights on this dataset.	The findings of this research paper are that YOLO-based networks are feasible for fall detection specifically in smart homes and it performs really well in several scenarios under controlled environment.	The Limitations of this paper were that its accuracy in real-world uncontrolled environments was not evaluated and issues like changeable light conditions and occlusions were a significant limitation.
Deep Neural Networks as Scientific Models[18]	R. M. Cichy, D. Kaiser	The method used in this research paper is to analyze the use of DNNS(Deep Neural Networks) in modeling cognitive functions, comparing them to traditional cognitive models.	The findings of the paper were that DNNs have successfully modeled multiple cognitive processes, thus gaining insight into working mechanism of human mind and also DNNs can capture nonlinear relationships in data.	The limitations are that the complicated nature of DNNS may pose a problem regarding some outputs, especially validation in cognitive models and the large quantities of data required for training DNNs which is not always available. There is a concern about the ability of DNNs to generalize results across different tasks.
Fall Detection Using Multiple Bioradars and Convolutional Neural Networks [19]	L. Anishchenko, A. Zhuravlev, and M. Chizh	The method used in this research paper is the combining of multiple bioradars with wavelet transform and deep learning techniques such as a well-trained CNN, AlexNet to detect falls.	The findings of the research paper are that multi-bioradar system can serve aa a minimalistic and view-independent fall detection method, achieving high accuracy and f1 scores of 99 %.	The limitations are that the dataset used to train the classifier was really small and obtained from young volunteers only suggesting that the performance of the system for elderly fall detection require further investigation.
A Fall Detection System Using k-Nearest Neighbor Classifier [20]	Chien-Liang Liu, Chia-Hoang Lee, and Ping-Min Lin	The authors use of silhouettes of human body for privacy protection and along with the statistical methods apply vertical projection histograms to reduce the effect of movements in upper limbs and then apply the k-Nearest Neighbor algorithm to classify the postures using height-to-width ratio of the bounding box of silhouette.	The findings were the proposed method was very effective in detecting falls using silhouette-based features and the k-NN classifier achieving a good accuracy of 84.44% in detecting fall and lying down events.	The limitations are that they do not provide the performance of the system with different types of user groups and real-world scenarios and further studies are required in order to analyze effectiveness of the system and overall applicability.
YOLO-Pose: Enhancing YOLO for Multi-Person Pose Estimation Using Object Keypoint Similarity Loss [21]	D. Maji, S. Nagori, M. Mathew, and D. Poddar	The authors propose YOLO-Pose, a heatmap-free method to perform multi-person 2-dimensional pose estimation and joint detection from the YOLO framework. YOLO-Pose detects a bounding box around each person, along with the pose, in one forward pass combining bottom-up as well as top-down approaches for efficient pose estimations.	The findings were that it performed better than any bottom-up method can be produced with a single forward pass with no need for augmentation-based tests as the YOLO-Pose attains state-of-the-art results on COCO validation set by providing 90.2% Average Precision at 50% Intersection over Union (AP50) and more importantly 90.3% AP50 on the test-dev set.	The limitations are that the study does not discuss the system's performance in the real-world scenarios and among different populations which is necessary to test the robustness of the system.
A New YOLO-Based Method for Real-Time Crowd Detection from Video and Performance Analysis of YOLO Models [22]	Mehmet Şirin Gündüz and Gültekin Işık	The authors in this paper propose to measure the size of a predefined region and count the number of people present in that area in real-time using YOLO-based models.	The results demonstrates the effectiveness of the proposed method in accurately detecting and counting individuals within a specific region in real-time video streams and that the YOLO-based approach is suitable for applications that require real-time crowd monitoring	The limitations of the study is that it does not test the robustness of the system under different conditions and addressing problems like varying light conditions, larger crowds and partial occlusions.
A Hybrid Human Fall Detection Method Based on Modified YOLOv8s [23]	Lei Liu, Yeguo Sun, Yinyin Li, Yihong Liu	This study presents a hybrid fall-detection approach integrating modified YOLOv8s with AlphaPose known known as HFDMIA-Pose. YOLOv8s model is used as object detector and uses SPD-Conv to preserve small object features and adds a small object detection layer and using BCIOU as loss function.	The findings show that hybrid approach have suited better for real time applications and the testing on LE2I dataset it was found to better perform fall detection and this approach was able to increase real-time processing with high fall detection rate.	The limitations are the performance of the system in case of multiple people or in complex fall scenarios and the long -term robustness of the system with uncontrolled real-world environments have not been explored.
Human Fall Detection using YOLO: A Real-Time and AI-on-the-Edge Approach [24]	R. S. Bhadoria, S. S. Sonawane, P. B. L. Meena, and R. S. Bichkar	The authors use YOLO for object detection to propose a real-time fall detection that efficiently detects falls by analyzing human posture in video feeds and is designed to run on edge devices and classifies as normal or fall.	The findings were that the proposed method performed well in various real-world scenarios and was able to process and accurately detect falls in real-time video streams. YOLO's precision and robustness make it ideal for systems where early responses are required.	The limitations were that the falls with crowd, multiple individuals and clutter were not evaluated and the dependency of having a clear and clean view of the person may interrupt its applicability.
SDES-YOLO: A High-Precision and Lightweight Model for Fall Detection [25]	Xiangqian Huang, Xiaoming Li, Limengzi Yuan, Zhao Jiang, Hongwei Jin, Wanghao Wu, Ru Cai, Meilian Zheng, Hongpeng Bai.	The methodology of this study is to design a SDES-YOLO lightweight system. The lightweight design helps it to work well on devices with very low computational resources making it ideal for edge computing applications and improving its scalability.	The findings were that SDES-YOLO compared to other fall detection models have better accuracy and faster processing, even in low-resource environments. This approach has been proven to be very accurate and suitable for real-time monitoring. This approach was validated for practical use cases such as ensuring safety in work environments and monitoring elderly in smart homes.	The limitations are that the generalization in diverse and complex scenarios like crowded and occluded scenes was not performed. The model should be validated in real-time uncontrollable environments and for future large-scale deployments there is a further prospect for research.

The dataset by Mastorakis et al. [13] was collected with a Microsoft Kinect camera which was fixed at 2014 cm height and proned towards the floor plane. The dataset contained information about 48 falls (sideways, leading, and rearward), 48 laying, 32 picking up, 32 sitting among other activities performed by eight subjects and two subjects to imitate elderly persons performed activities in slow motion.

Other datasets are also reported but not publicly available. A dataset was presented by Auvinet et al. [14] containing falls and other normal activities were mimicked with an eight camera arrangement and only one person mimicked falls. Various kinds of falls were recorded like loss of balance, backwards fall, forward fall, fall while sitting down. Other daily living activities like sitting, laying, crouching, moving up and down, standing up were also collected. Microsoft Kernel was used in recollecting a very ample dataset which was conferred in [15]. The dataset contained data of around 16 subjects in old age settings collecting 454 (445 mimicked and 9 real) falls and other activities like standing, lying down and sitting positions.

B. YOLO ALGORITHM AND ITS VERSIONS

The YOLO (You Only Look Once) algorithm is a real-time object detection system that has revolutionized computer vision. It is known for its high speed and accuracy in predicting objects in real-time image and video streams. The advancements in YOLO version from the very original YOLOv1 to the latest YOLOv11 has progressively increased its performance specifically in terms of detection speed and accuracy. YOLOv11 was introduced on 30 September, 2024 with advanced features making it an ideal candidate for real-time fall detection [5]. Unlike previous versions, with low resources it performs better than its predecessors. It comes with models including Bounding Box Models, segmentation (-seg), Pose Estimation (-pose), and Classification (-cls). All these models come in five different sizes of Nano (n), Small (s), Medium (m), Large (l) and Extra Large (x) [26].

Model	Filenames	Task	Inference	Validation	Training	Export
YOLO11	yolo11n.pt yolo11s.pt yolo11m.pt yolo11l.pt yolo11x.pt	Detection	✅	✅	✅	✅
YOLO11-seg	yolo11n-seg.pt yolo11s-seg.pt yolo11m-seg.pt yolo11l-seg.pt yolo11x-seg.pt	Instance Segmentation	✅	✅	✅	✅
YOLO11-pose	yolo11n-pose.pt yolo11s-pose.pt yolo11m-pose.pt yolo11l-pose.pt yolo11x-pose.pt	Pose/Keypoints	✅	✅	✅	✅
YOLO11-obb	yolo11n-obb.pt yolo11s-obb.pt yolo11m-obb.pt yolo11l-obb.pt yolo11x-obb.pt	Oriented Detection	✅	✅	✅	✅
YOLO11-cls	yolo11n-cls.pt yolo11s-cls.pt yolo11m-cls.pt yolo11l-cls.pt yolo11x-cls.pt	Classification	✅	✅	✅	✅

FIGURE 2: An illustration of YOLOv11's model types and sizes. [27]

C. CHALLENGES IN FALL DETECTION

Fall detection systems face various challenges including: **Misclassification**: As seen with the model in [9], the incorrect classification of fall activities as non-fall activities and non-fall activities as fall activities. Fall Detection systems face a significant problem of misclassifying activities similar to falls such as laying down, sitting as fall activities.

Real-time Detection: In fall detection systems real-time processing and detection is very critical as timely intervention can make the difference between life and death. Achieving real-time performance without comprising on its accuracy remains a significant challenge in vision-based systems.

Generalization: Fall detection systems must generalize well across various uncontrollable real-time environments and varying video conditions such as varying lighting, camera angles and backgrounds. Overfitting on specific datasets can lead to poor performance on the unseen and real-time data.

D. PATTERNS OBSERVED FROM THE RESEARCH PAPER

Many papers utilize YOLO as the base model or a hybrid between YOLO and other frameworks such as YOLOv8 [17], YOLO-Pose [21], and SDES-YOLO [25] due to its performance, efficiency, accuracy and real-time performance in object detection tasks.

Focus on Lightweight Models for Real-Time Applicability: There is a constant emphasis on using lightweight models to enable deployment on edge devices with low computational resources, such as low resource IOT devices in smart homes.

Validation on Specific Datasets: Most of the studies validate their models on specific custom datasets or LE2I dataset [23] that simulate falls in specific and controlled environments. These datasets typically include fall events and daily activities.

Accuracy and Speed Improvements: Many methods highlight improvements among processing speed and accuracy, showing the capability to handle real-time fall detection, which is important for time-sensitive situations such as elderly care.

Incorporation of Pose Estimation: Several studies incorporate pose estimation or body keypoint detection to analyze postures of human to detect actions such as identifying falls, moving beyond simple bounding box detection. [21], [23], [24]

Applications in Practical Domains: The primary use cases that these studies focus on are smart homes, public safety and elderly care, where fall detection systems are very important.

Limitations in Generalization and Real-World Robustness and Applicability: A recurring theme of limitation across studies was the lack of evaluation in uncontrallable, real-world environments such as dynamic and complex backgrounds, crowded and occluded scenes and varying lighting

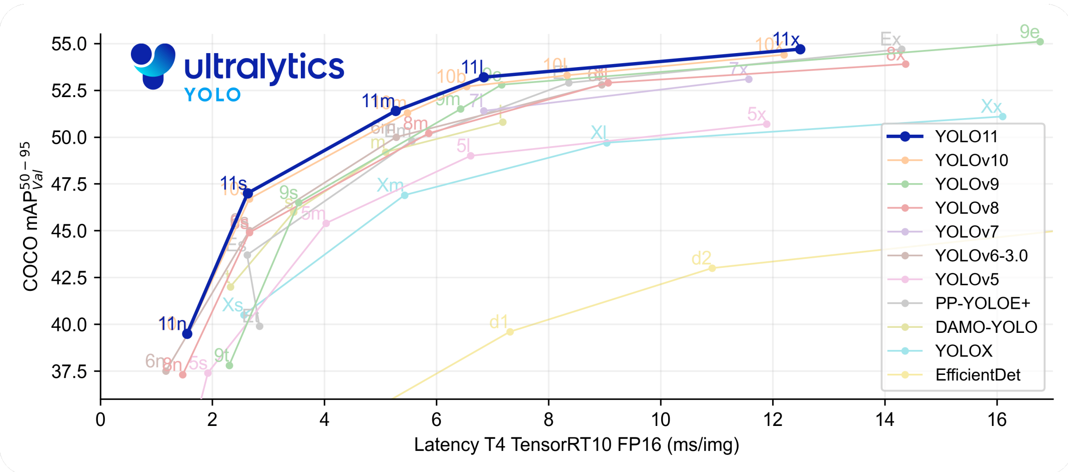

FIGURE 3: A comparison between performance of YOLOv11 and its predecessors. [26]

conditions. Generalization among diverse populations like different cultural settings is also very rarely exposed.

Dependency on Dataset and Experimental Settings: Performance is more often than not tailored with specific datasets and generalization among various conditions and dataset is not tested and thus creates a gap in understanding the robustness and applicability of the models.

Data Challenges: Many proposed methods and studies face difficulties like limited diversity of datasets, small training datasets and huge dependency on unoccluded and clear view of individuals.

Large-Scale Deployments and Scalability: Studies more often than not lacke exploration into scalablity for large-scale deployments such as handling multipe individuals at the same time or having multi camera setup.

Future Research Directions: We can leverage the performance, processing speed and accuracy of YOLOv11 to address the issues like testing in uncontrolled, real-world scenarios. Improving generalizability and scalability for larger deployments and address the issues of lighting, camera variations, occlusions and handling multiple individuals simultaneously.

E. COMMON METHODOLOGIES OBSERVED ACROSS THE RESEARCH PAPER

The most commonly used based model among most studies was YOLO (You Only Look Once). Be it YOLOv8 [23] or hybrid approaches like YOLO-Pose [21], and SDES-YOLO [25]. These models are adapted for specific tasks like fall detection, pose estimation and lighteight deployments.

Keypoint Detection and Pose Estimation: Several methods integrate keypoint detection [16] or pose estimation [21]to analyze posture of humans and distinguish between fall and other normal activities. Techniques like keypoint vectorization or Seq2Seq [16] architecture are used to classify activities and predict future poses.

Lightweight Model Design: There is a great focus on lightweight models [25] and design in these studies so they can optimized for real-time performance and edge computing and making deployment possible of models even in resource-constrained devices like IOT sensors and cameras.

Hybrid Approaches: Many studies use YOLO as a object detector [24] and combine the base model with additional components like wavelet transforms to extract features [19] or using k-Nearest Neighbour (k-NN) Classifier for classification [20], or even using CNN (Convolutional Neural Networks) and AlexNet for enhanced feature processing [19].

Custom Preprocessing Techniques: To enhance efficiency of models, some papers employ custom preprocessing techniques such as vertical projection histograms to simplify data, vectorization of keypoints [16], and Silhouette analysis for feature privacy preservation [20].

Use of Specialized Datasets: Models are trained and validated on specialized datasets like LE2I dataset [23] or custom dataset which contains simulated falls and other activities.

Separate Training of Modules: Some methods train specific modules such as pose prediction and fall classification separately, followed by fine-tuning to enhance overall performance of the models.

Deployment on Edge-Devices: Multiple studies focus on designing models suitable for deployment on edge devices to avoid reliance on cloud computing and ensure real-time applicability [25].

Real-Time Classification: A shared goal among all studies is the real-time detection and classification of falls and other activities in video streams.

Privacy Preservation: As privacy is really important. So Privacy is a consideration in certain papers, achieved through:

1. Non-contact sensors like bioradars [19].
2. Silhouette-based features [20],
3. Avoidance of raw RGB data in favour of keypoint or pose data [17].

Evaluation Metrics: Most researches evaluate their performance mainly on four metrics that are:

1) Accuracy
2) Precision
3) Recall
4) F1 Score

These metrics are computed using validation and test datasets.

TABLE 4. Summary of evaluation metrics.

Metric	Description	Formula
Accuracy	Quantifies the degree of correctness in the predictions made by a model.	$\frac{tp'+tn'}{tp'+fp'+tn'+fn'}$
Precision	Computes the ratio of true positive predictions to the total number of positive predictions generated by the model.	$\frac{tp'}{tp'+fp'}$
Recall	Computes the ratio of correctly predicted positive cases to the total number of genuine positive cases in the dataset.	$\frac{tp'}{tp'+fn'}$
F1-score	The harmonic mean of recall and precision, offering an optimal blend of recall and precision for unified evaluation of the model's effectiveness.	$\frac{2*Precision*Recall}{Precision+Recall}$

FIGURE 4: Illustration of evaluation metrics and their relationships.

F. IMPORTANCE OF THE RESEARCH

The research project aims to address the above challenges and create a real-time vision-based fall detection systems leveraging the object detection model of YOLOv11, fine-tuning it on the LE2I fall based dataset to improve real-time detection and classification accuracy. The LE2I dataset provides us a really strong and robust foundation for fall-related activity recognition and by modifying it to our need for better performance, this research aims to push the boundaries of the current and already present fall detection systems.

The use of YOLOv11, combined with the LE2I dataset, represents a leap towards creating a system capable of accurately predicting falls in dynamic, real-world environments. The ability to detect falls in real-time video streams and differentiate between fall and non-fall activities is really crucial and important for enhancing the safety of the vulnerable populations, including elderly care and healthcare settings.

III. METHODOLOGY

The methodology of our system is divided into four phases, with each phase addressing a really crucial part of the system's development and ensuring a comprehensive approach for solving the problem. The phases are: The **first phase** focused on doing thorough research, gaining and increasing our knowledge and to decide the methodology and approach for our system. It included thorough research related to fall-detection systems and different models and approaches used in fall-detection systems. The **second phase** included preparation of data that included first dividing videos into frames and then annotating the frames for our YOLO model. YOLO annotations play a very important role in YOLO training. This phase also included choosing the best 2 versions of YOLO for our system that are YOLOv8 and YOLOv11 models. The **third phase** included the actual training part where training on the annotated dataset was done using YOLOv8's and YOLOv11's nano versions with about 100 epochs for each of them. The **fourth and final phase** included testing and validating the system, evaluating their performances and finally documenting the results in the final project report.

TOOLS USED

The following tools were used for our system:

Hardware: NVIDIA T4 Tensor Core GPU

Software: We used Python (Programming Language), YOLO library (via Ultralytics module), OpenCV (for video and image processing), PyTorch (dependency of Ultralytics), Matplotlib (for data visualization), and Kaggle environment (for data processing, model training and experimentation) [30].

Dataset: We used publicly available LE2I dataset using a subset of video frames for it that depicted various scenarios including fall and non-fall activities.

FIGURE 5: Illustration of falsely annotated frame depicting non-fall activity as fall.

PHASE I: DATA COLLECTION AND PREPARATORY RESEARCH

The initial approach/methodology of our system was to use a model pre-trained on the UCF-101 Dataset [7], which contains 101 action categories and then further train on UP-Fall Detection dataset [29] and detect fall as an anomaly so our system can detect all other 101 action categories as non-fall and detect fall as an anomaly. But after thorough research we got to know about the YOLOv8 (You Only Look

Once version 8) object detection model and its high accuracy and real-time performance which was very crucial for our system. So we modified our idea to use YOLOv8 as a pre-trained model and fine-tune it to detect fall activities. Then after doing further research we learnt about the LE2I dataset which included videos of fall and non-fall activities which are annotated frame by frame as 0 (non-fall) and 1 (fall).

PHASE II: DATA ANNOTATIONS AND CHOOSING THE BEST MODEL

So our approach was finally decided to use the YOLOv8 and YOLOv11 models on a subset of LE2I dataset as LE2I is a very vast dataset. After deciding the approach we did some research and found out that work has already been done using YOLOv8 on a subset of LE2I dataset in [9], but it was classifying fall as non fall activities and non-fall as fall activities. We trained YOLOv8 nano version again on that dataset to ensure the results and it still classified fall as non-fall activity and non-fall activity as fall. Model's performance was still great with high accuracy but the data was incorrectly annotated. So to improve this we used the annotated dataset used in [9] and then correctly annotated it as because of the false annotations the earlier model in [9] was predicting non-fall activities as falls and falls as non fall activities.

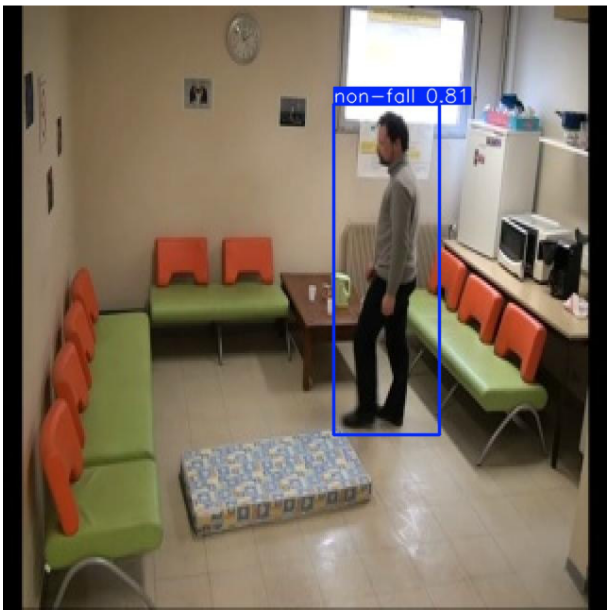

FIGURE 6: Illustration of correctly annotated frame depicting non-fall activity as non-fall.

Structure of the Dataset:The dataset was divided into further sets of testing, training and validation sets, with directories for images and their corresponding annotations.

Annotations: Bounding boxes for fall and non fall events were already included in this dataset, formatted in YOLO11 compatible .txt files which contained the following:

1. Class id: an integer representing the object class.
2. X center: The normalized x-coordinate of the center of bounding box. (a value between 0 and 1)
3. Y center: The normalized y-coordinate of the center of bounding box. (a value between 0 and 1)
4. Width: The normalized width of the bounding box. (a value between 0 and 1)
5. Height: The normalized height of the bounding box. (a value between 0 and 1)

PHASE III: TRAINING ON YOLOV11

YOLO models itself have many versions namely Nano (n), Small (s), Medium (m), Large (l), and Extra Large (x). So considering the time we had and the computational resources we had, we chose nano variant of YOLOv11 to train our model for its balance between performance and computational efficiency.

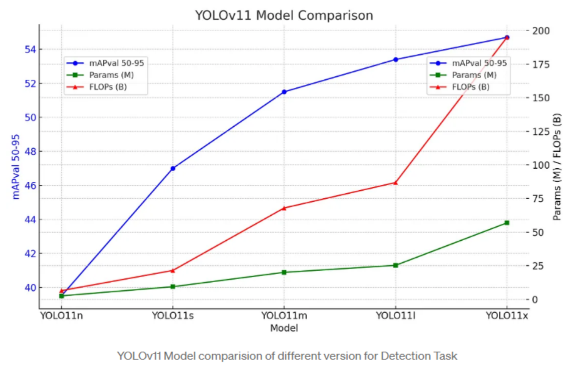

FIGURE 7: Comparison of different YOLOv11 models for Detection Tasks.

The Training parameters were as follows:
- **Learning rate**: 0.001.
- **Optimizer**: Adam.
- **Image size**: 640 × 640 pixels.
- **Batch size**: 16.
- **Epochs**: 100.
- **Early stopping** was disabled to ensure thorough training.

Initially I enabled early stopping with patience of 10 but then the model stopped training after 14 epochs only it was correctly performing classification but it was classifying non-human such as almirahs and tables as objects predicting falls and non-falls or it as the box loss, class loss, and the distribution focal loss was very high. For this purpose I disabled early stopping to ensure thorough training as the box, class and distribution focal loss was gradually decreasing.

PHASE IV: EVALUATION AND VALIDATION

In the last and final phase of our project, our model was validated using the validation set using metrics such as precision, recall, **mAP50**(Mean Average Precision at 50 percent Intersection over Union (IoU)) and **mAP50-95**Mean Average Precision at 50 percent to 95 percent Intersection over Union (IoU) (in steps of 5 percent).

It performed really well with the overall scores of **0.991, 0.985, 0.991 and 0.792 for precision, recall, mAP50, and mAP50-95** respectively.

```
                      from  n    params  module                                       arguments
  0                     -1  1       464  ultralytics.nn.modules.conv.Conv             [3, 16, 3, 2]
  1                     -1  1      4672  ultralytics.nn.modules.conv.Conv             [16, 32, 3, 2]
  2                     -1  1      6640  ultralytics.nn.modules.block.C3k2            [32, 64, 1, False, 0.25]
  3                     -1  1     36992  ultralytics.nn.modules.conv.Conv             [64, 64, 3, 2]
  4                     -1  1     26080  ultralytics.nn.modules.block.C3k2            [64, 128, 1, False, 0.25]
  5                     -1  1    147712  ultralytics.nn.modules.conv.Conv             [128, 128, 3, 2]
  6                     -1  1     87040  ultralytics.nn.modules.block.C3k2            [128, 128, 1, True]
  7                     -1  1    295424  ultralytics.nn.modules.conv.Conv             [128, 256, 3, 2]
  8                     -1  1    346112  ultralytics.nn.modules.block.C3k2            [256, 256, 1, True]
  9                     -1  1    164608  ultralytics.nn.modules.block.SPPF            [256, 256, 5]
 10                     -1  1    249728  ultralytics.nn.modules.block.C2PSA           [256, 256, 1]
 11                     -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 12                [-1, 6]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 13                     -1  1    111296  ultralytics.nn.modules.block.C3k2            [384, 128, 1, False]
 14                     -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 15                [-1, 4]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 16                     -1  1     32096  ultralytics.nn.modules.block.C3k2            [256, 64, 1, False]
 17                     -1  1     36992  ultralytics.nn.modules.conv.Conv             [64, 64, 3, 2]
 18               [-1, 13]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 19                     -1  1     86720  ultralytics.nn.modules.block.C3k2            [192, 128, 1, False]
 20                     -1  1    147712  ultralytics.nn.modules.conv.Conv             [128, 128, 3, 2]
 21               [-1, 10]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 22                     -1  1    378880  ultralytics.nn.modules.block.C3k2            [384, 256, 1, True]
 23           [16, 19, 22]  1    431062  ultralytics.nn.modules.head.Detect           [2, [64, 128, 256]]
YOLO11n summary: 319 layers, 2,590,230 parameters, 2,590,214 gradients, 6.4 GFLOPs
```

FIGURE 8: YOLOv11n model summary. [26]

For fall class, it had scores of **0.988, 0.983, 0.987 and 0.797 for precision, recall, mAP50, and mAP50-95** respectively.

For non-fall class, it had scores of **0.993, 0.987, 0.994 and 0.786 for precision, recall, mAP50, and mAP50-95** respectively.

Results were also visualized through Image and Video Annotations. Test images were processed through the trained model, and predictions were visualized as annotated bounding boxes (depicting fall or non-fall activity) on the images.

Unseen videos from the LE2I dataset [9] and some real-life videos were also processed through the model and each frame was annotated with bounding boxes (depicting fall or non-fall activity with certain confidence levels).

At last, model's best weights, training logs, and annotated outputs both images and videos were compressed and archived for future use and deployment if needed.

IV. RESULTS

The results section include the Fall Detection system's findings in an organized and simple manner. Three models were trained one YOLOv8 on the wrong annotated dataset, second YOLOv11 on the correctly annotated dataset for 14 epochs (due to callback/patience) and thirdly the last model YOLOv11 for 100 epochs to thoroughly train the model and lessen the box, class and distributed focal loss so model accurately only predict human as objects not other things. **Box Loss**: This loss measure how accurately the model predicts bounding box size and location for detected objects compared to ground truth.

Class Loss: This loss evaluates how well model classifies objects into their correct classes.

DFL (Distributed Focal Loss): By focusing on fine-grained localization, it is used to improve the precision of bounding box regression.

FIGURE 9: Annotated frame depicting fall activity with bounding boxes in real-time.

Model (Epochs)	Box Loss	Class Loss	DFL Loss	Precision (P)	Recall (R)	mAP50	mAP50-95
YOLOv8 (100, inverse annotations)	0.6757	0.2763	0.9359	0.986	0.980	0.992	0.795
YOLOv11 (14 epochs)	1.09	0.6379	1.105	0.982	0.972	0.989	0.745
YOLOv11 (100 epochs)	0.6937	0.2848	0.9399	0.987	0.985	0.990	0.790

TABLE 1: Evaluation Metrics Comparison for YOLOv8 and YOLOv11 Models

CHOOSING THE BEST MODEL

By comparing the performance of all the three models, we can choose YOLOv11 for its good results and less computational resources usage as it will be beneficial for real-time performance and fast processing. This model can easily be integrated into the edge device such as IOT devices in smart homes.

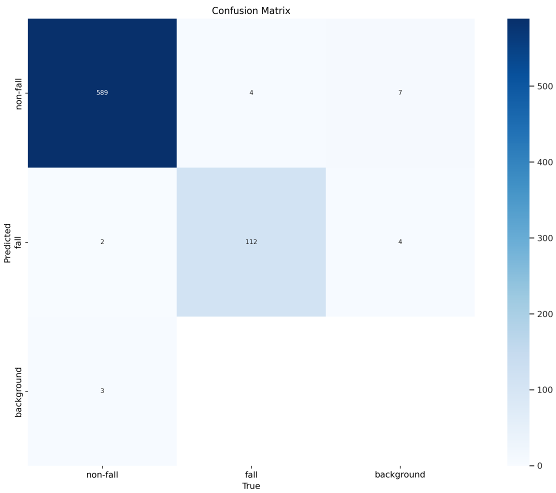

FIGURE 10: Confusion Matrix of YOLOv11 100 epoch model.

As we can see the in Figure 10, the performance of our model for both classes is superb making it an ideal system for fall detection.

As we can see in Figure 11, the model at high confidence levels is almost equal to 1 showing the model's capability of precisely predicting only true positives even at high confidence levels.

As we can see the in Figure 12, the model at high confidence levels of 0.8 and greater does not perform that well indicating that when predicting a class with more than 80 percent confidence model tends to predict fall activity as non-fall activity (false negative).

As we can see the in Figure 13, as the f1 score is the harmonic mean of recall and precision it also like recall curve take a sharp leap downwards indicating model's inability to classify false negatives correctly at high confidence levels. Model performs almost perfectly at around till 78-80 percent

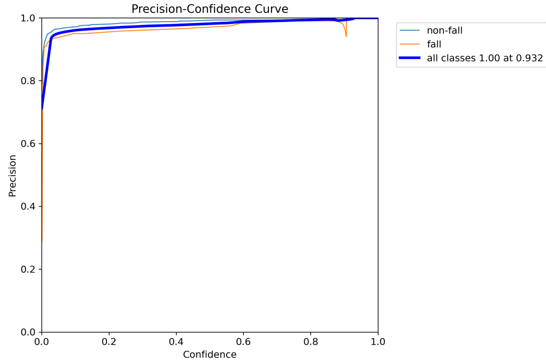

FIGURE 11: Precision Curve of YOLOv11 100 epoch model.

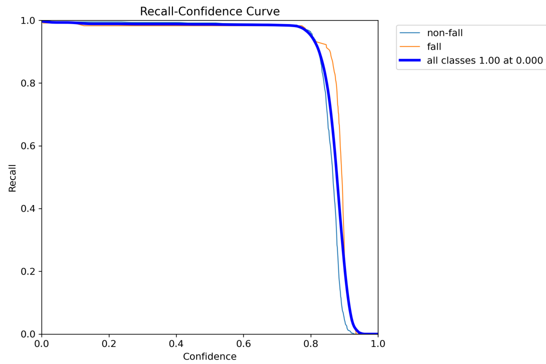

FIGURE 12: Recall Curve of YOLOv11 100 epoch model.

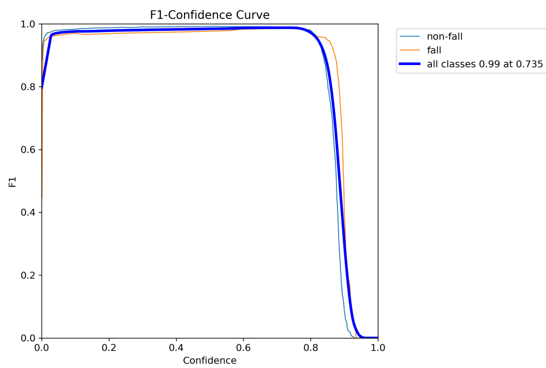

FIGURE 13: F1 Curve of YOLOv11 100 epoch model.

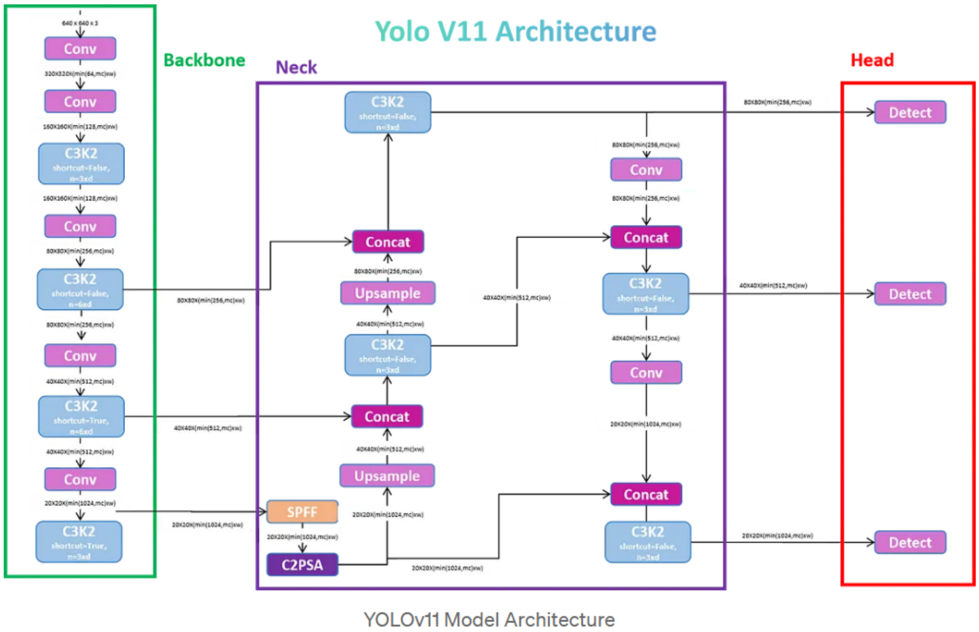

FIGURE 14: Architecture of YOLOv11 Model [28]

confidence then almost fails miserably at confidence levels higher than 80 percent.

V. CONCLUSION

This study efficiently handles the gaps and limitations of the previous fall detection specifically performance with high accuracy in real-life uncontrollable scenarios and creating a model that can easily and efficiently work on edge devices for large-scale deployment. There is always a need for improvement as nothing is perfect, there are some limitations of YOLOv11 model as the loss specifically the distributed focal loss is still high. We can try further training on more epochs to see if DFL (Distributed Focal Loss) improves or not or we need a hybrid approach for more better performance. Future work can explore incorporation multi-modal sensor data, real-time deployment and additional data augmentation techniques to improve the model's performance further and reduce latency in real-world environments.

APPENDIX

A. DATASET DESCRIPTION

The LE2I fall Detection dataset, used for training and eveluating the YOLOv11 model and contains frame by frame annotated video sequences designed specifically for fall detection in real-life scenarios with each frame labelled with 0 (non-fall) and 1 (fall) class along with the bounding boxes indicating absence or presence of falls. The dataset features videos captured in different situations ensuring robust model training for various angles. In addition some of the videos of the LE2I dataset [9] were also processed and used to evaluate our model's performance on unseen data.

B. MODEL CONFIGURATION AND HYPERPARAMETERS

The following hyperparameters were used for YOLOv11 model:

- **Learning Rate**: 0.001
- **Batch Size**: 16
- **Epochs (YOLOv11)**: 14 and 100 epochs were used to assess model performance at different training stages.
- **Image Size**: 640x640 pixels.
- **Optimizer**: Adam optimizer

The model was trained with sole focus on efficient and highly accurate fall detection.

C. MODEL PERFORMANCE EVALUATION

Table 1 shows the evaluation metrics for both the YOLOv8 and YOLOv11 models. YOLOv11 shows consistent performance with notable improvement in box loss, class loss and distributed focal loss over time.

D. PRECISION AND RECALL CURVES

Figures 11 and 12 show the precision and recall curves of YOLOv11 model after 100 epochs of training with precision curve implying that model achieves a near-perfect precision at higher confidence levels. Similarly, recall curve demonstrates that even at low confidence levels model is able to correctly and effectively identify fall activities.

E. LIMITATIONS AND FUTURE WORK

While the performance of our YOLOv11 model is very good in fall detection tasks, there are certain gaps and limitations that can be addressed in future works:

- **False Negatives**: The model occasionally miss falls, especially in cases of occlusion and unclear view of the person.
- **Model Size**: The models could further benefit from optimization to reduce computational overhead for real-time applicability especially on edge devices.
- **Generalization**: Additional datasets and real-world evaluation can help improve the model's robustness across distinct environments and video qualities.

F. CODE AVAILABILITY

The code for training, validating and evaluating the model as well as applying the model on real-time video streams is available at the project's Kaggle repository at [30].

ACKNOWLEDGMENT

We would like to acknowledge Dr. Shaista Habib [32] Program Head of Artificial Intelligence at Department of Artificial Intelligence [31], University of Management and Technology in Lahore [33], for her crucial support and assistance during the project's development and this write-up. Her encouragement and expertise made a compelling contribution to the success and completion of our study.

We are also thankful to UMT's Department of Artificial Intelligence [31] for providing infrastructure specifically the HPC (Highe Performance computing) Lab and assistance. Special acknowledgment for our peers and mentors for constant support throughout our study.

Finally, we thank the creators of LE2I Fall Detection Dataset [**?**] for making vast collection of videos containing fall and non-fall activities publicly available which served basis of our study. This study could not have been completed without their collective expertise and contribution to research and academic communities.

REFERENCES

[1] M. Mubashir, L. Shao, and N. L. Seed, "A survey on fall detection: Principles and approaches," Neurocomputing, vol. 100, pp. 144–152, 2013. [Online]. Available: https://api.semanticscholar.org/CorpusID:207102846

[2] World Health Organization (WHO), "Falls," [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/falls

[3] Centers for Disease Control and Prevention (CDC), "Falls Are Leading Cause of Injury and Death in Older Americans," [Online]. Available: https://www.cdc.gov/falls/facts.html

[4] I. Charfi, W. P. Marnane, and A. K. Bourke, "A vision-based fall detection system using the LE2I fall detection dataset," in Proc. IEEE Int. Conf. Image Process. (ICIP), 2013, pp. [insert specific page numbers if available].

[5] Ultralytics, "YOLOv11 Model Documentation," 2025. [Online]. Available: https://docs.ultralytics.com/models/yolo11/. [Accessed: Jan. 25, 2025].

[6] T. L. D. Nguyen, "FallDataset-IMVIA," Kaggle, 2021. [Online]. Available: https://www.kaggle.com/datasets/tuyenldvn/falldataset-imvia. [Accessed: Jan. 25, 2025].

[7] J. K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," CRCV, 2012. [Online]. Available: https://www.crcv.ucf.edu/data/UCF101.php. [Accessed: Jan. 25, 2025].

[8] YOLOv8, "YOLOv8 Official Website," 2025. [Online]. Available: https://yolov8.com/. [Accessed: Jan. 25, 2025].

[9] LE2I-AHLAM, "LE2I-AHLAM Model," Roboflow, 2025. [Online]. Available: https://universe.roboflow.com/le2iahlam/le2i-ahlam/model/1. [Accessed: Jan. 25, 2025].

[10] X. Ma, H. Wang, B. Xue, M. Zhou, B. Ji, and Y. Li, "Depth-based human fall detection via shape features and improved extreme learning machine," IEEE Journal of Biomedical and Health Informatics, vol. 18, no. 6, pp. 1915–1922, 2014.

[11] X. Zhang, Y. Watkins, and G. T. Kenyon, "International Symposium on Visual Computing," 2014.

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

[13] G. Mastorakis and D. Makris, "Fall detection system using Kinect's infrared sensor," Journal of Real-Time Image Processing, vol. 9, pp. 635–646, 2014.

[14] E. Auvinet, C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau, "Multiple Cameras Fall Dataset," University of Montréal Technical Report 1350, 2019. [Online]. Available: http://www-labs.iro.umontreal.ca/~labimage/Dataset/technicalReport.pdf. [Accessed: Apr. 16, 2019].

[15] E. E. Stone and M. Skubic, "Fall detection in homes of older adults using the Microsoft Kinect," IEEE Journal of Biomedical and Health Informatics, vol. 19, no. 1, pp. 290–301, 2014.

[16] M. Hua, Y. Nan, and S. Lian, "Falls prediction based on body keypoints and seq2seq architecture," in Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, pp. 0–0, 2019.

[17] B. Luo, "Human fall detection for smart home caring using YOLO networks," International Journal of Advanced Computer Science and Applications, vol. 14, no. 4, 2023. [Online]. Available: https://www.scienceandinformation.org. [Accessed: Jan. 26, 2025].

[18] R. M. Cichy and D. Kaiser, "Deep neural networks as scientific models," Trends in Cognitive Sciences, vol. 23, no. 4, pp. 305–317, 2019.

[19] L. Anishchenko, A. Zhuravlev, and M. Chizh, "Fall detection using multiple bioradars and convolutional neural networks," Sensors, vol. 19, no. 24, p. 5569, 2019.

[20] C.-L. Liu, C.-H. Lee, and P.-M. Lin, "A fall detection system using k-nearest neighbor classifier," Expert Systems with Applications, vol. 37, no. 10, pp. 7174–7181, 2010.

[21] D. Maji, S. Nagori, M. Mathew, and D. Poddar, "YOLO-pose: Enhancing YOLO for multi-person pose estimation using object keypoint similarity loss," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2637–2646, 2022.

[22] M. Ş. Gündüz and G. Işık, "A new YOLO-based method for real-time crowd detection from video and performance analysis of YOLO models," Journal of Real-Time Image Processing, vol. 20, no. 1, p. 5, 2023.

[23] L. Liu, Y. Sun, Y. Li, and Y. Liu, "A hybrid human fall detection method based on modified YOLOv8s and AlphaPose," Scientific Reports, vol. 15, no. 1, p. 2636, 2025.

[24] A. Raza, M. H. Yousaf, and S. A. Velastin, "Human fall detection using YOLO: A real-time and AI-on-the-edge perspective," in 2022 12th International Conference on Pattern Recognition Systems (ICPRS), pp. 1–6, 2022.

[25] X. Huang, X. Li, L. Yuan, Z. Jiang, H. Jin, W. Wu, R. Cai, M. Zheng, and H. Bai, "SDES-YOLO: A high-precision and lightweight model for fall detection in complex environments," Scientific Reports, vol. 15, no. 1, p. 2026, 2025.

[26] Ultralytics, "YOLOv11 Supported Tasks and Modes," Ultralytics Documentation. [Online]. Available: https://docs.ultralytics.com/models/yolo11/#supported-tasks-and-modes. [Accessed: Jan. 26, 2025].

[27] Z. Shariff, "YOLOv11 is Officially Out: What You Need to Know," Medium, Jan. 2025. [Online]. Available: https://medium.com/@zainshariff6506/yolov11-is-officially-out-what-you-need-to-know-6738c5d25be1. [Accessed: Jan. 26, 2025].

[28] N. Rao, "YOLOv11 Explained: Next-Level Object Detection with Enhanced Speed and Accuracy," Medium, Jan. 2025. [Online]. Available: urlhttps://medium.com/@nikhil-rao-20/yolov11-explained-next-level-object-detection-with-enhanced-speed-and-accuracy-2dbe2d376f71. [Accessed: Jan. 26, 2025].

[29] L. Martínez-Villaseñor, H. Ponce, J. Brieva, E. Moya-Albor, J. Núñez-Martínez, and C. Peñafort-Asturiano, "UP-Fall Detection Dataset: A Multimodal Approach," Sensors (Basel), vol. 19, no. 9, p. 1988, Apr. 2019, doi: 10.3390/s19091988. PMID: 31035377; PMCID: PMC6539235.

[30] S. Burhan Ahmed, "Fall Detection using YOLOv11," Kaggle, [Online]. Available: https://www.kaggle.com/code/syedburhanahmed/fall-detection-yolov11/edit/run/218837785. [Accessed: Jan. 27, 2025].

[31] Department of Artificial Intelligence, Lahore, SST, “Department of Artificial Intelligence,” *University of Management and Technology (UMT)*, [Online]. Available: https://sst.umt.edu.pk/ai/home.aspx. [Accessed: Jan. 27, 2025].

[32] Dr. Shaista Habib, “Dr. Shaista Habib,” *University of Management and Technology (UMT)*, [Online]. Available: https://sst.umt.edu.pk/ai/Academics/Faculty/Dr-Shaista-Habib.aspx. [Accessed: Jan. 27, 2025].

[33] University of Management and Technology (UMT), “University of Management and Technology (UMT),” [Online]. Available: https://www.umt.edu.pk/. [Accessed: Jan. 27, 2025].

[34] L. Bouzid, A. F. Chahabi, H. D. Fathy, “LE2I Labs,” LE2I Laboratory, Université de Bourgogne, France. Available: https://www.nature.com/nature-index/institution-outputs/france/laboratory-of-electronics-information-and-image-le2i/5371ab8e140ba04767000000. [Accessed: Jan. 27, 2025].