# Section 1: Project Definition

## 1.1 Project Overview:

Motivated by the growing demand for solutions that improve safety, this project aims to address the problem of delayed medical attention by detecting falls and possible risks quickly. The image dataset contains various scenarios of falls occurring in different settings and environments.

The solution could be used to help several domains, including:

- Elderly care homes
- High-risk workplaces like construction sites.
- Public spaces such as parks, streets, or shopping centres.

## 1.2 Problem Statement:

There is a need for a reliable system that can promptly identify falls and potential injuries in various settings, such as elderly care homes, high-risk workplaces, and public spaces. Without such a system, caregivers and emergency medical services may not be able to respond quickly. This can lead to delayed medical attention, increased risk of complications, and reduced safety for individuals in these settings. For this, the goal of this project is to build a Fall Detection model that can be integrated into smart home systems, monitoring services, and security cameras to detect falls and alert relevant parties in a timely manner, improving the quality of care and safety.

## 1.3 Metrics:

The most commonly used metric in object detection is the mean average precision mAP, which I will be using to evaluate the model's performance. mAP formula is based on the following sub metrics: Confusion Matrix, Intersection over Union (IoU), Recall ,and Precision.

True Positive (TP) —t0he classifier predicted positive when the truth is indeed positive, that is, a detection for which IoU ≥ α.

False Positive (FP) — this is a wrong positive detection, that is, a detection for which IoU < α.

False Negative (FN) — this is an actual instance that is not detected by the classifier.

True Negative (TN) — the classifier does not predict the label when the truth is negative.

Precision (P) — the True Positives out of total model's predictions.

P= TP/(TP+FP)

Recall (R) — the True Positives out of total positive ground truths.

R= TP/(TP+FN)

Average Precision (AP) — the area under the precision-recall curve.

Mean Average Precision or (mAP) — the average of AP over all detected classes.

# Section 2: Analysis

## 2.1 Data Exploration:

The dataset consists of images of falling people in different environments and scenarios. There is one class called (Fall-Detected) and the dataset doesn't contain any background images. As it is an object detection problem, the labels consist of the bounding box boundaries of the object. All examples in the dataset are stretched to 640x640 image size and Auto-Orientation applied.

The dataset is randomly divided into 88% Training Set, 8% Validation Set, and 4% Testing Set.

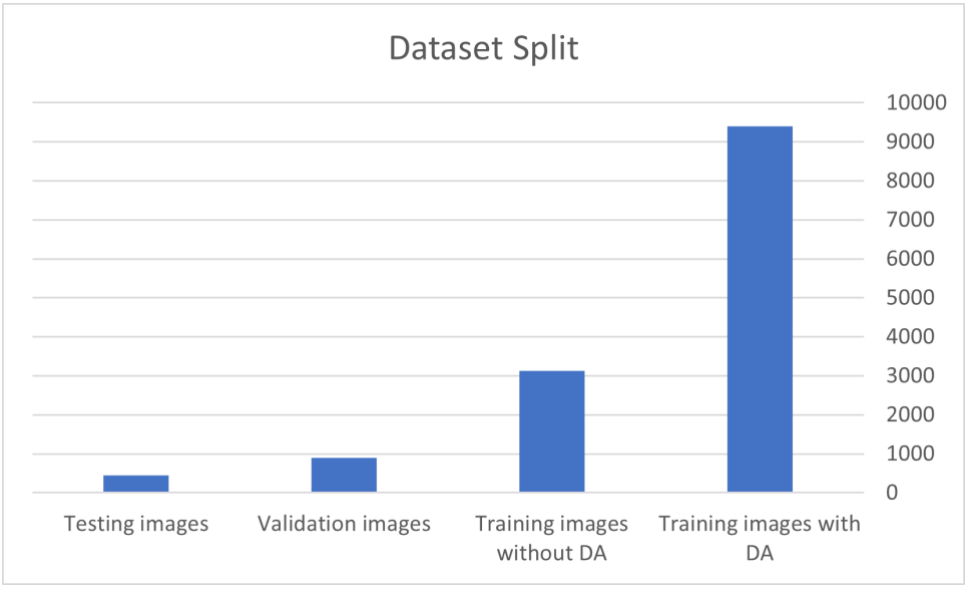
Also, every training image has three augmented copies, the list of used augmentations is:

- Flip: Horizontal
- Crop: 0% Minimum Zoom, 20% Maximum Zoom
- Rotation: Between -12° and +12°
- Shear: ±2° Horizontal, ±2° Vertical
- Grayscale: Apply to 10% of images
- Hue: Between -20° and +20°
- Saturation: Between -20% and +20%
- Brightness: Between -20% and +20%
- Exposure: Between -20% and +20%
- Blur: Up to 0.75px
- Cutout: 5 boxes with 3% size each

The following figure shows some images in the fall detection dataset where the fall detection model will be very helpful solution.

## 2.2 Data Visualization:

The dataset consists of 3133, 899, and 450 images for Training, Validation, and Testing respectively. For training each image has 3 augmented copies resulting 9399 images as shown in the figure below.



# Section 3: Methodology

## 3.1 Data Preprocessing:

The dataset comes already pre-processed, all training examples have the same size and each one has copies where several augmentations where applied.

## 3.2 Implementation:

For modelling, I used the state-of-the-art (SOTA) model in object detection Yolov8 which is the latest version of You Only Look Once (YOLO) family. Yolov8 is built on cutting-edge advancements in deep learning and computer vision, offering unparalleled performance in terms of speed and accuracy. It is easily adaptable to different hardware platforms, from edge devices to cloud APIs which is important for the problem I am trying to solve. Moreover, YOLOv8 has a Python interface provided by Ultralytics which allows to quickly implement the object detection model. Yolov8 comes with several Pre-trained Weights for object detection, as shown in the following table.

| Model | size (pixels) | $mAP^{val}$ 50-95 | Speed CPU ONNX (ms) | Speed A100 TensorRT (ms) | params (M) | FLOPs (B) |
|---|---|---|---|---|---|---|
| YOLOv8n | 640 | 37.3 | 80.4 | 0.99 | 3.2 | 8.7 |
| YOLOv8s | 640 | 44.9 | 128.4 | 1.20 | 11.2 | 28.6 |
| YOLOv8m | 640 | 50.2 | 234.7 | 1.83 | 25.9 | 78.9 |
| YOLOv8l | 640 | 52.9 | 375.2 | 2.39 | 43.7 | 165.2 |
| YOLOv8x | 640 | 53.9 | 479.1 | 3.53 | 68.2 | 257.8 |

For training the yolov8 model in the fall detection dataset, I used the pretrained nano model because the dataset size is not large enough for larger model depth. Another reason is that nano model is the fastest, which is important to avoid the Google Colab disconnection. I trained the model for 100 epochs on 1 GPU using Google Colab, which is a free cloud service that supports free GPU.

The following picture shows examples from the first training batch.

## 3.3 Refinement:

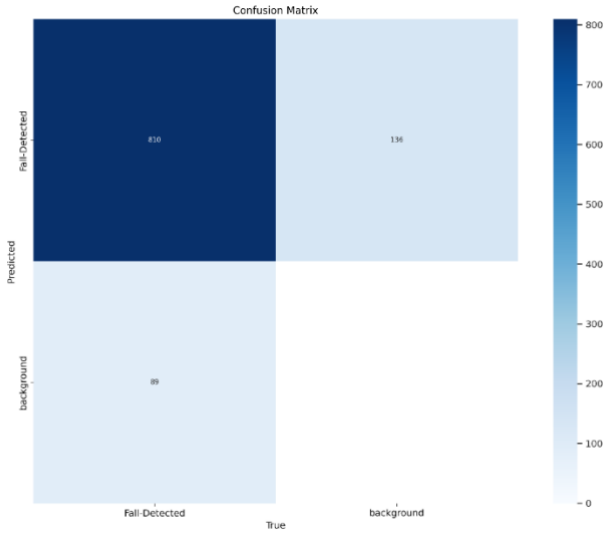
Although I assigned the training epochs to be 100, the training early stopped at epoch 85. This is because I assigned the patience to be equal to 10, which means the training would early stop if there were no observable improvement for 10 consecutive epochs. The best results observed at epoch 76 which I used for this project. For other hyperparameters I used the default values.

# Section 4: Results

## 4.1 Model Evaluation and Validation:

After training the model for 85 epochs, the model best performance was at the epoch 76, where the mAP@50-95 (the average mAP over different IoU thresholds, from 0.5 to 0.95) was equal to 0.56677.

The ultralytics yolov8 provides the confusion matrix plot after the training is completed, which shows that the model has predicted 136 False positive and 89 False negatives as shown in the figure below.




The following pictures shows the GT in the first validation batch and the model predictions.

Ground Truth

Model Predictions

### 4.2 Justification:

The model performance is not good enough to solve the problem, but it is a useful baseline. There are several reasons for the results. First, the resources limitation, the google colab disconnects for long sessions which means I can't train the model for longer epochs. Also, the GPU in google colab is only free for limited hours. Second, the dataset contains very difficult examples. As you can see in the training and validation batch above. In validation batch, there are some fallen people without label. In training batch, some examples have very bad lightening conditions, and the object wasn't clear at all.

## Section 5: Conclusion

### 5.1 Reflection:

The process used in this project can be summarized in the following steps:

1- Searching for the best dataset for the problem.

2- The dataset was downloaded using roboflow.

3- Exploring the dataset to ensure it satisfies the problem needs.

4- Searching for the best model for object detection.

5- Prepare the environment and installing the requirements.

6- Choose the size of the model and the number of epochs to match the resources limitations.

7- The classifier was trained using nano model weights for 100 epochs with defaults hyperparameters.

8- Assess the results and plan for improvement when the resources are not limited.

I find that step 5 was the most difficult, I had to train the model multiple times because the google colab disconnects after epoch 40.

The most interesting part was searching for a relevant dataset and exploring the data.

### 5.2 Improvement:

- I would prepare my own dataset and use data labelling platform.

- I will limit the scope of the problem to make the dataset easier.

## Section 6: References


https://colab.research.google.com/

https://docs.ultralytics.com/modes/train/

https://universe.roboflow.com/roboflow-universe-projects/fall-detection-ca3o8/dataset/4

https://learnopencv.com/mean-average-precision-map-object-detection-model-evaluation-metric/

https://www.v7labs.com/blog/mean-average-precision