Elderly Fall Detection for Smart Home Caring Using YOLOv8

Aanchal Verma[1], Harsh Verma[2], and R.K. Saket[1,*]

ARTICLE INFO

Article history:
Received: 24 September 2023
Revised: 15 December 2023
Accepted: 27 December 2023
Online: 20 June 2024

Keywords:
Deep learning
YOLO
Smart home
Fall detection

ABSTRACT

Climate variability and climate change cause of drought events. Drought is a significant impact of economic, environment and social. The suitable methods in inspecting and monitoring provide useful information that can be used to build up prevention and mitigation planning from drought impacts. The study purposes were to find out the suitable drought index and its trend for the Northern of Thailand. The 10 stations meteorological data during 1951-2020 were used for the drought indices calculation. The indices were Standardized Precipitation Index (SPI), Deciles index and Moisture Available index (MAI) at 1, 3, 6 month timescales and monthly Palmer Drought Severity Index (PDSI). It found that PDSI was the most appropriate index. Consequently, the Mann-Kendal test was used to investigate the trend of PDSI for long-term (40 years) and short-term (20 years) periods. There was no trend of drought intensity for long-term period but found in 4 stations for short-term period. The PDSI can be used for drought monitoring and prediction by using numerical weather prediction products.

1. INTRODUCTION

Today, the healthcare industry's top priority is to prevent falls, which impair people's physical and emotional health, especially the elderly. A significant portion of society has entered an era of ageing, and this problem is gaining more attention among researchers. Many technologies are currently being researched to prevent it. As people age or have health issues like cardiovascular disease, reason to certain medication or muscle instability, falls become more common. Support for the physical and mental health will be essential if elders wish to continue living independently.

A trustworthy fall detection and emergency aid system is therefore required by several stakeholders in the healthcare ecosystem. Some systems have used gyroscopes, inertial sensors, and barometers to detect falls. In these applications, sensors are generally employed to detect sudden changes in a person's activities. These sensors can be added to cell-phones, smart bracelets, or smart necklaces, to make them "wearable" gadgets. The fact that these devices need to be attached to the subject's body and that monitoring is sensor-dependent is their biggest drawback. As a result, individuals without sensors are invisible during any situations that resemble falls. There are numerous options for these products in the market. These items' main drawbacks are their high price and extensive hardware component requirements. Despite these sensor-based methods for fall detection, computer vision can be seen as a less intrusive field. The researchers have presented a few image/video datasets and investigated a wide range of methodologies.

In 2025, the population will consist of over 300 million individuals aged 60 and above, representing 20.7 percent of the total population [1]. The growing population of elderly individuals, particularly those living alone, has raised significant concerns about their daily safety. Both their families and society at large are increasingly worried about ensuring the well-being of elderly individuals who are living independently. Falls are identified as the second-leading cause of unintentional injuries and fatal accidents, making them a critical concern for the safety of elderly people. Unfortunately, falls constitute the leading cause of injury-related fatalities in this particular age group [2]. Hence, investigating geriatric fall detection holds great importance due to its substantial societal consequences [3].

Since there are various methods for detecting falls, let's begin with conventional signal-based methods. The three most common types of fall detection systems currently in use are those based on wearable sensor devices, computer vision, and sensors deployed in the surrounding [4]. The senior activity area should have a variety of monitoring devices to collect data such as sound, vibration, and pressure. Based on the sensors deployed in various environmental circumstances, this data will be used to determine whether a fall has occurred.

This method's detection area has several restrictions, the

[1]*Department of Electrical Engineering, Indian Institute of Technology (Banaras Hindu University), Varanasi-221005 (UP) India.*
[2]*Vindhya Institute of Technology and Science, Satna (MP), India*.
**Corresponding author:* R.K. Saket; E-mail: rksaket.eee@iitbhu.ac.in.

sensors are susceptible to interference from the surroundings, and the detection accuracy is low [5], [6]. Wearable sensor devices are needed for fall detection depending on the old person's waist, limbs, chest, or back. These devices incorporate sensors such magnetic needles, gyroscopes, and accelerators. The movement of the elderly during a specific time period is then detected and analysed using the sensor data, which can help identify whether a fall has occurred. Although this technology offers easy installation and a high rate of detection, its effectiveness relies on consistent wear, which can disrupt the daily lives of the elderly. Opting not to wear the device hinders swift status determination, and the necessity for frequent recharging adds to the inconvenience [7], [8].

Traditional algorithms analyze video frame indicators, such as the person's angle from the vertical and the rate at which their height varies, and then estimate if a fall will occur under particular predetermined circumstances [9]. Although there are some clear drawbacks, this oversimplifies the issue. The complexity of actual falls cannot be simplified to merely determining whether certain features are below or over a given threshold. The features obtained through machine learning methods are similar in quality to those derived from traditional techniques. These characteristics are employed for the purpose of training a machine learning classifier, that determines whether or not a fall has occurred, as opposed to having a predetermined criterion.

Methods for identifying targets through the use of Convolutional Neural Networks (CNNs) generally fall into two primary categories [10]. A two-stage detection approach is involved in the first category, where the process of locating and recognizing the target is divided into distinct phases. The first method within this category, known as Region-Convolutional Neural Network (R-CNN), demonstrated less-than-ideal results and did not meet the demands for processing in real-time. To overcome this drawback, subsequent innovations like Faster R-CNN [11] and Fast R- CNN [12] were introduced, yet they remained inadequate in satisfying the need for real-time performance. The second category encompasses one-stage detection methods, which streamline the target's localization and identification into a single step. "You Only Look Once" (YOLO) and the single shot multi-box detector (SSD) series are well-known examples of this methodology. In 2020, Lu et al. [13] implemented a novel approach to detect falls by integrating a spatial visual attention mechanism utilizing long short-term memory (LSTM) with a three-dimensional convolutional neural network (3D CNN).

Zhang et al. [14] devised a technique to predict the likelihood of a person experiencing a fall by analyzing spatial and temporal changes in body posture. Their approach incorporated a diagram depicting the evolution of human behavior in terms of time and space. In 2021, Zhu et al. introduced a method involving a deep vision sensor combined with a CNN [15]. The CNN was trained on three-dimensional body posture data to predict falls, although its real-time performance was not notably exceptional. Cao et al. [16] suggested an approach for detecting falls utilizing a combination of deep learning and motion characteristics. Their approach combined the deep features extracted by a CNN with human motion characteristics to identify fall incidents. For recognizing human subjects, the method employed the YOLO version3 (YOLOv3) algorithm. Notably, the YOLO algorithm has undergone updates, with the most recent version being YOLO version8 (YOLOv8). Compared to its predecessors YOLOv8 offers significantly improved detection speed alongside enhanced precision and a smaller model size.

2. METHODOLOGY

2.1 YOLO

The world of object detection experienced a revolutionary transformation when the YOLO algorithm was introduced. This ground breaking advancement ushered in the era of real-time object recognition by executing just one forward pass of the neural network. Object detection was reimagined by YOLO as a regression task, with probabilities provided for the classes of detected objects. Redmon et al. [11] unveiled this remarkable achievement in 2016, making it possible to conduct seamless, real-time end-to-end training while maintaining outstanding average precision. Fig. 1 illustrates the YOLO architecture.

The YOLO framework divides the input image into a grid of size SxS. When the central point of an object aligns with a grid cell, that particular cell becomes responsible for detecting the object. Within each grid cell, B bounding boxes and corresponding confidence scores are forecasted. These confidence scores indicate the model's certainty regarding both the presence of an object and the accuracy of the prediction. To compute confidence, the intersection over union (IOU) between actual and predicted boxes is calculated. In grid cells where no objects are present, confidence scores remain at zero.

Each bounding box prediction encompasses five key elements: w, h, x, y and confidence. The (x, y) coordinates signify the center of the box relative to the edges of the grid cell. The height (h) and width (w) are defined relative to the entire image. The confidence prediction serves as a representation of the IOU between the predicted and actual boxes.

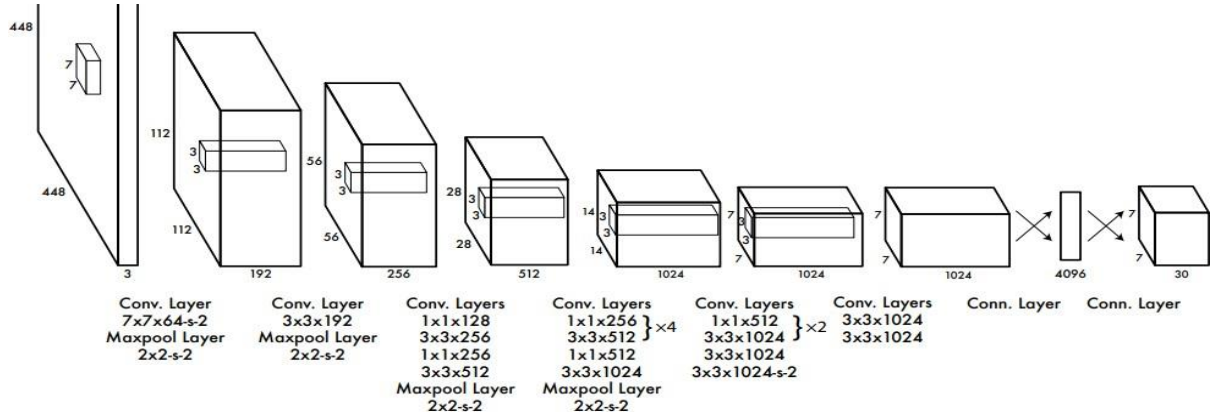

Fig. 1. YOLO Architecture [11].

Table 1: Brief description YOLO variants

S/N	YOLO Variant	Improvement	Results
1	YOLOv1 (Redmon et 2016) [11]	The single shot detector addresses and resolves the issue of simultaneously drawing boundary boxes and identifying object classes.	Higher accuracy and speed in contrast to a two-stage object detection system, such as Faster R-CNN
2	YOLOv2 (Redmon Farhadi, 2018)[17]	Successive advancements have been made to enhance batch normalization, enabling higher-resolution detection, and adopting anchor boxes for improved results.	Architectural reduction, improved high resolution picture detection, and faster, more accurate detection
3	YOLOv3 (Redmon Farhadi, 2018)[17]	Adding connections to the backbone network layers, adding the objectness score to bounding box prediction, and predictions at three different levels of granularity.	Improves detection of smaller objects
4	YOLOv4 (Alexey et al., 2020)[18]	Improved feature aggregations, incorporating a bag of complimentary techniques, including mosaic augmentations, and embracing the mish activation function.	Improved training efficiency and accuracy, as well as performance excellence and accessibility
5	YOLOv5 (Nepal & Eslamiat, 2022)[19]	Reduced network parameters are employed, combining Cross Stage Partial Network (CSPNet) in the architectural head and PANet in the neck, with the inclusion of residual structures and auto-anchoring. Furthermore, mosaic augmentations are incorporated.	Inference on batch, video feed, and webcam ports is incredibly simple to train. Using and transferring weights with ease. Lighter and faster than prior models.
6	YOLOv6 (Chuyi et al., 2022)[20]	Rep-PAN Neck and EfficientRep Backbone are the neck designs and new network backbone. Different features are separated from the final head by decoupling the network head.	Improvement in small object detection, anchor-free model training. less flexible and stable than YOLOv5.
7	YOLOv7 (Wang et al., 2022)[21]	Extended Efficient Layer Aggregation Network (E-ELAN) layer aggregation, trainable freebies, and a reduction of 35% in network parameters.	Improved accuracy and speed, and simplified training
8	YOLOv8 (Ram, et al., 2023)[22]	Introduces Darknet-53 backbone network, which is quicker and more precise than the one used in YOLOv7. And uses larger feature map for better detection	Higher accuracy and speed compared to other algorithm.

The foundational structure of YOLO draws inspiration from the GoogLeNet image classification model. This architecture boasts 24 convolutional layers, followed by 2 fully connected layers. In place of GoogLeNet's inception modules, 1x1 reduction layers and 3x3 convolutional layers are integrated. Over time, the YOLO architecture underwent substantial refinements, introducing algorithms and procedures aimed at bolstering accuracy, reducing network size, and amplifying detection speed. Subsequent versions like YOLOv5, YOLOv7 and YOLOv8 further refined the

architecture to achieve better detections. These versions incorporate various advancements summarized in Table 1. YOLOv5, YOLOv7, and YOLOv8 introduce enhancements in precision, accelerated detection speeds, and streamlined network architectures.

2.2 Improvement of YOLO

The primary distinctions between the YOLOv1, YOLOv2, YOLOv3, YOLOv4, and YOLOv5 architectures, according to Nepal and Eslamiat (2022), are that YOLOv1 utilises the softmax function and that YOLOv2 has a higher resolution classifier and is more accurate and efficient than YOLOv1. This is as a result of the CNN of YOLOv2 having a batch normalisation layer added. To extract features from the input image, YOLOv3 employs Darknet53 as its primary backbone, which has a greater efficiency and detection performance. Multi-object categorization, or the ability for an object to simultaneously belong to many categories, is available in YOLOv3. To determine the likelihood that an input image corresponds to a particular label, YOLOv3 swaps out the softmax function with an independent logistics function. Additionally, YOLOv3 employs the 2-class entropy loss for each category, which lessens the computational complexity introduced by softmax functions. The CSPDarknet53 network, which combines Darknet53 and CSP network, serves as the backbone of the YOLOv4 architecture. Higher accuracy, greater object detection efficiency, and fewer hardware requirements are all features of YOLOv4. Focus structure, using CSP-Darknet53 as its skeleton, is used by YOLOv5. In YOLOv5, the Focus layer makes its debut. In the YOLOv3 algorithm, the Focus layer takes the place of the previous three layers. Incorporating a Focus layer offers several advantages, including reduced Compute Unified Device Architecture (CUDA) memory usage, a smaller layer size, and improved backward and forward propagation efficiency.

Comparatively, YOLOv5 demonstrates impressive speed while being approximately 90 percent lighter in terms of size compared to YOLOv4.

YOLOv6 adopts various advancements such as the EfficientRep Backbone, RepVGG Style structure, SimOTA algorithm, SIoU bounding box regression loss function and Rep-PAN Anchor-free paradigm.

In contrast, YOLOv7 distinguishes itself by outperforming all current object detection systems in respect of both precision and speed across a frame rate spectrum spanning from 5 to 160 frames per second. Specifically, YOLOv7 uses rep-Path Aggregation Networks (PAN) and it attains the top accuracy rating of 56.8% among real-time object detectors running at 30 frames per second or higher on the GPU V100.

YOLO's seventh version has realized a noteworthy enhancement in real-time object detection precision while maintaining the same inference costs. Furthermore, it has managed to decrease the computational requirements for achieving a cutting-edge real-time object detector by roughly 40%, all while enhancing both detection and inference speed.

2.3 YOLOv8

Ultralytics, the creators of YOLOv5, introduced YOLOv8 [23] in January 2023. YOLOv8 offers a range of five scaled versions: YOLOv8l (large), YOLOv8m (medium), YOLOv8s (small), YOLOv8x (extra-large), and YOLOv8n (nano). The capabilities of YOLOv8 include object identification, tracking, segmentation, pose estimation, and classification, among other computer vision tasks. Fig.2 provides a comprehensive visualization of the YOLOv8 architecture [23].

By introducing modifications to the CSPLayer, now referred to as the C2f module, YOLOv8 maintains its structural resemblance to its predecessor, YOLOv5. One significant improvement introduced in YOLOv8 involves the incorporation of the C2f module, known as the cross-stage partial bottleneck with two convolutions. This module effectively blends advanced features with contextual insights, leading to an increased level of precision in object detection. YOLOv8 demonstrates efficiency in autonomously handling objectness, classification, and regression tasks through the implementation of a decoupled head and an anchor-free model. This modular strategy contributes to improved accuracy and task specialization.

In the output layer of YOLOv8, the objectness score, denoting the likelihood of object presence within a bounding box, is activated using the sigmoid function. For assessing class probabilities, YOLOv8 adopts the softmax function, providing insights into the likelihood of objects belonging to different classes. To handle loss functions, YOLOv8 uses the CIoU (Complete Intersection over Union) [24] and DFL (Distribution Focal Loss) methods for bounding box loss, while it employs binary cross-entropy for classification loss. This combination of loss functions is particularly effective in enhancing object detection accuracy, particularly for detecting smaller objects.

Furthermore, YOLOv8 introduces an innovative model called YOLOv8-Seg, designed for semantic segmentation tasks. This model incorporates a CSPDarknet53 feature extractor as its backbone, coupled with a C2f module. Positioned after the C2f module, two segmentation heads are employed to forecast semantic segmentation masks for input images. YOLOv8-Seg preserves the architecture of five detection modules along with a prediction layer, aligning with the configuration of YOLOv8's detection heads.

The YOLOv8-Seg model has demonstrated performance in a variety of benchmarks for both object detection and semantic segmentation, striking a balance between speed and efficiency. YOLOv8 can be conveniently installed as a PIP package or executed through a user- friendly command-line interface (CLI). It offers seamless integrations for tasks such as labeling, training, and deployment.

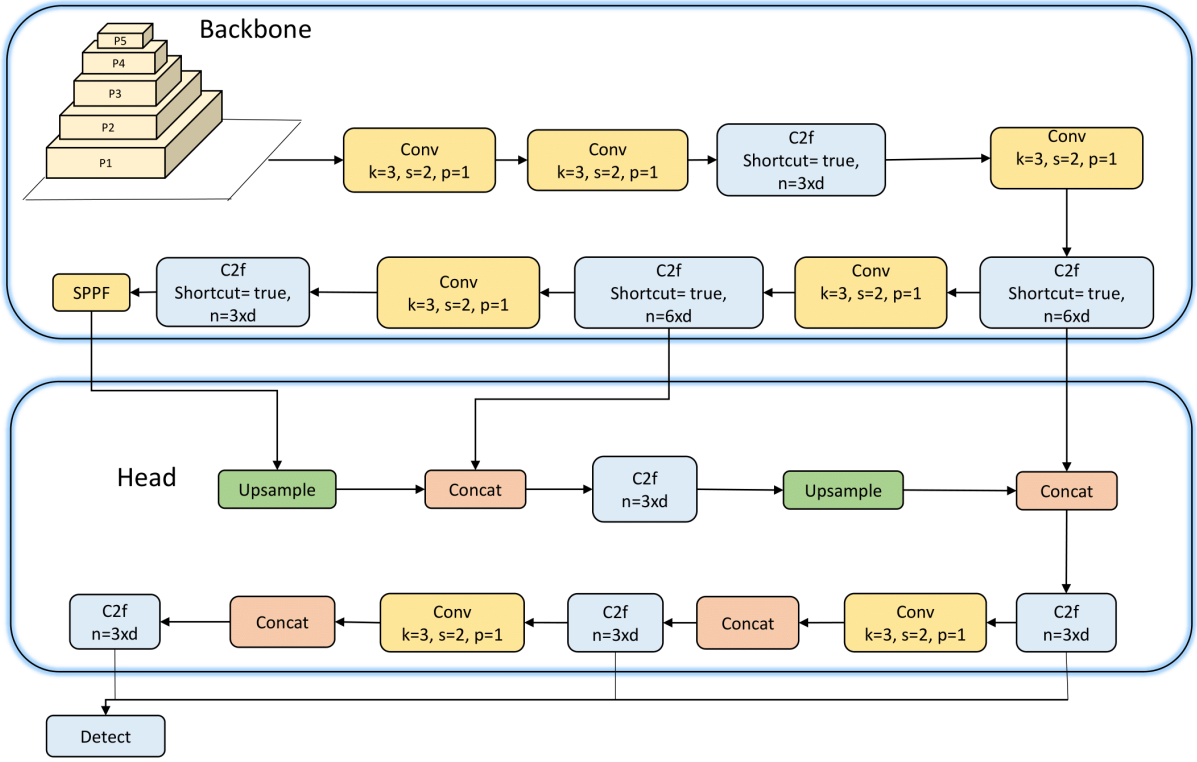

Fig. 2(a). YOLOv8 Architecture.

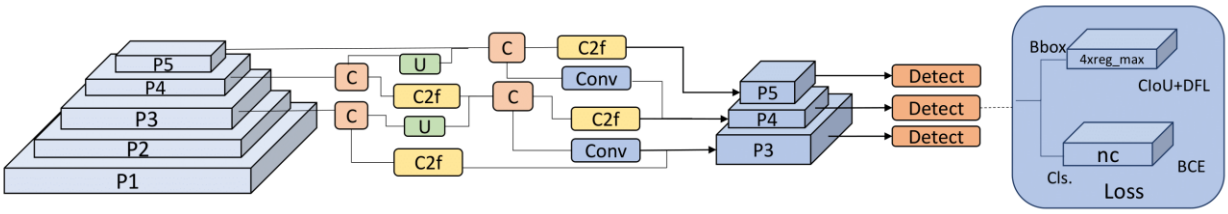

Fig. 2(b). YOLOv8 BCE Loss, Distribution Focal Loss and CIoU Loss.

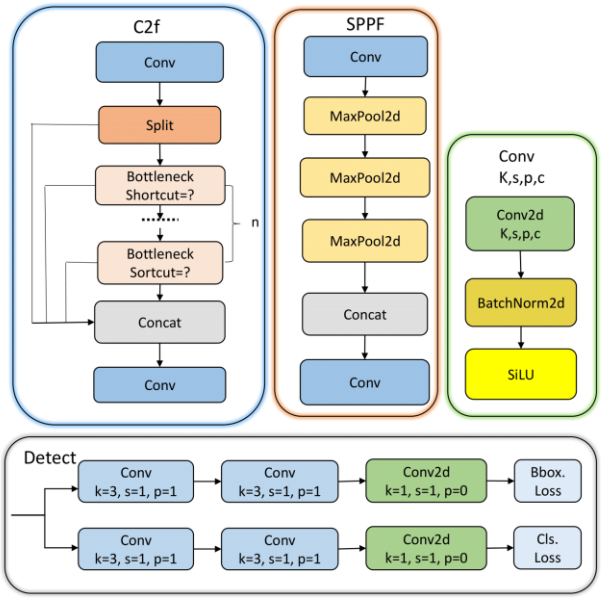

Fig. 2(c). C2f, SPPF, Conv, Detection block details.

In the assessment conducted on the MS COCO dataset, YOLOv8x garnered a remarkable performance score of 53.9% while being evaluated at an image size of 640 pixels. This achievement notably outperforms YOLOv5, which secured a score of 50.7% using the identical input dimensions. An additional highlight is the operational speed of YOLOv8x, clocking in at an impressive 280 frames per second (FPS) on an NVIDIA 100 GPU enhanced with TensorRT.

2.4. Dataset

A fall detection dataset, comprising images and corresponding labels, was compiled by collecting photographs from diverse sources. The classifications was employed in this study pertain to "normal" and "falling" scenarios. The dataset encompasses two main subdirectories: "Training" (containing 374 photos) and "Validation" (comprising 111 images), both located within the "images" directory. These subdirectories serve distinct purposes in the dataset. Specifically, text files containing labels for individual images are present in this directory. Fig.3 illustrates various instances showcasing our dataset's composition.

Fig. 3. Sample images from dataset.

2.5. Google Colab

Google Colab comes equipped with pre-installed libraries like TensorFlow, Keras, and PyTorch, which are widely employed for deep learning applications such as object detection [25]. Moreover, it offers complimentary access to GPUs and TPUs, which can considerably expedite the training process of YOLOv8 models.

In this study, Google Colab was employed, leveraging the availability of robust GPUs without cost. The model was trained for 100 epochs, with each training iteration consisting of a batch size of 16.

2.6. Performance Metrices

Precision (P) in object detection assesses the accuracy of a model's predictions by measuring the proportion of correctly identified objects among all objects predicted as positive. Conversely, Recall (R) gauges the model's capability to detect all objects present in an image. The P-R Curve demonstrates the interplay between recall and precision across different confidence thresholds. The PR curve's integral signifies the Average Precision (AP) score, providing a graphical representation of the model's effectiveness.

The AP is a measurement that merges recall and precision across varying confidence thresholds. The Mean Average Precision (mAP) defines the mean value of AP scores computed for distinct classes within a multi-class object detection scenario. The Intersection over Union metric quantifies the degree of intersection between the actual bounding box of an object and a predicted bounding box.

The F1 Score, computed as the harmonic average of recall and precision, provides a balanced metric that takes into account both false negatives and false positives. It serves as a comprehensive metric for evaluating both recall and precision simultaneously. A higher F1 score signifies an improved balance between these two metrics. The mathematical representations of various performance metrics are provided below.

$$\text{mAP} = \frac{1}{\text{N}}\sum_{\text{i}=1}^{\text{N}} \text{AP}_\text{i} \tag{1}$$

The notation AP_i signifies the Average Precision score corresponding to class i, with N representing the overall count of classes.

$$P = \frac{True\ Positive\ (Tp)}{False\ Positive\ (Fp) + True\ positive(Tp)} = \frac{Tp}{all\ detections} \tag{2}$$

$$R = \frac{Tp}{False\ Positive\ (Fp) + True\ Negative(Tn)} = \frac{Tp}{all\ detections} \tag{3}$$

$$\text{F}_1 = \frac{PXR}{P+R}\text{x2} \tag{4}$$

$$Accuracy = \frac{Tp+Tn}{Fp+Fn+Tp+Tn} \tag{5}$$

where, T_p is the overall count of obstacles that were correctly detected, F_p is the overall count of obstacles that were incorrectly detected objects, and F_n is the total number of obstacles are not correctly detected, T_n is the number of correctly classified negative instances.

In summary, these performance metrics employed in YOLO provide a quantitative evaluation of the model's accuracy, efficiency, and overall detection capabilities.

3. EXPERIMENTAL RESULTS AND PERFORMANCE ANALYSIS

Table 2 offers a summary of the training times for various iterations of the YOLOv8 model. Notably, YOLOv8n emerges as the fastest to train, with an approximate duration of 0.430 hours. This expedited training time may position

YOLOv8n as an attractive option for situations where swift model deployment is crucial, provided its performance metrics do not markedly lag behind other models.

Table 2: Training time of different YOLOv8 models

Model	Training Time
YOLOv8l	0.900 hours
YOLOv8x	1.433 hours
YOLOv8m	0.624 hours
YOLOv8s	0.510 hours
YOLOv8n	**0.430 hours**

After YOLOv8n, YOLOv8s steps in with a training duration of 0.510 hours. Although it requires a bit more time compared to YOLOv8n, it still stands as a relatively short training period when compared to the other models. Positioned as the intermediate model, YOLOv8m requires a notably longer training period, clocking in at approximately 0.624 hours.

On the other hand, YOLOv8l takes even more time to train, approximately 0.900 hours. In comparison, YOLOv8x, which has the lengthiest training duration, demands around 1.433 hours. These prolonged training periods are probably due to the more complex model architectures, potentially leading to improved performance metrics.

Table 3: Performance metrices of different models

Model	P	R	mAP50	F1-Score
YOLOv8x	0.775	0.832	0.869	0.80
YOLOv8l	0.835	**0.917**	**0.95**	0.87
YOLOv8m	0.886	0.783	0.882	0.83
YOLOv8s	**0.943**	0.881	**0.953**	**0.91**
YOLOv8n	0.897	0.914	**0.952**	0.90

Table 3 provides a comparison of performance metrics for different YOLOv8 variants on the validation dataset. YOLOv8n attains a precision of 0.897 and a recall of 0.914. These metrics collectively result in an F1-score of 0.90 and an mAP@50 of 0.952, showcasing its comprehensive proficiency in object detection and localization tasks. In contrast, YOLOv8m, which is a comparatively more complex model than YOLOv8s and YOLOv8n, fails to outperform the other models, including YOLOv8s and YOLOv8n. YOLOv8m attains an mAP@50 of 0.882 and an F1 score of 0.83.

Moving towards the YOLOv8x model, despite the longer training time, it does not deliver superior results. It achieves mAP@50 of 0.869 and an F1 score of 0.80. Fig. 4 illustrates a comparison of various metrics using a bar graph. Both YOLOv8s and YOLOv8l perform quite well, with YOLOv8l achieving an mAP@50 of 0.95, recall is 0.917 and an F1 score of 0.87, and YOLOv8s achieving an mAP of 0.953 and an F1 score of 0.91. Fig. 5, 6, 7, and 8 illustrate the progression of metrics throughout the validation epochs.

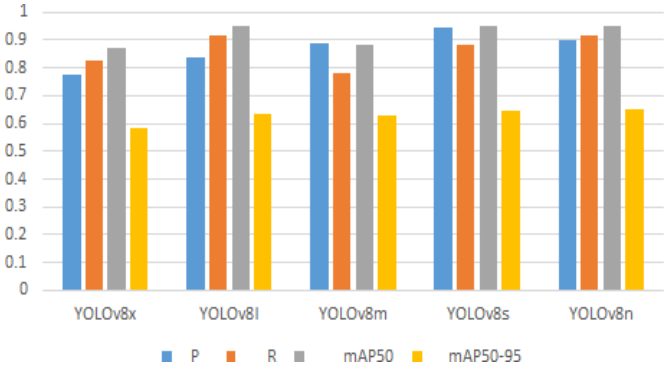

Fig. 4. A bar chart comparing various performance metrics between different YOLOv8 models.

Table 4: Best recall of models

Model	Best Recall	Epoch
YOLOv8l	**0.913**	126
YOLOv8x	0.861	126
YOLOv8m	0.848	128
YOLOv8s	**0.91**	126
YOLOv8n	**0.91**	111

In the early epochs, there is noticeable variability in metrics across all models, especially in recall and precision. F1-score and mAP@50, on the other hand, demonstrate comparatively less fluctuation.

Achieving stability poses a significant challenge across the entire range of models, particularly with YOLOv8l and YOLOv8x encountering the most pronounced difficulties. It becomes evident from figures that a semblance of stability begins to emerge only after reaching approximately epoch120. Nevertheless, this stability predominantly applies to the metrics of mAP@50 and F1-score. Changes in recall and precision endure throughout the latter phases of training, with certain models exhibiting a decline in performance as they approach the conclusion of the training procedure.

Concerning convergence, the YOLOv8l and YOLOv8x models display a more gradual convergence pattern across all metrics, with recall and precision showing particularly noticeable delays in attaining consistent performance. In contrast, the lighter models, namely YOLOv8m, YOLOv8s, and YOLOv8n, exhibit quicker convergence, greater overall stability, and no worrisome signs of performance decline over epochs. These observations shed light on the intricate connection between stability and convergence within the assessed models. The need for a greater number of training iterations to achieve optimal performance becomes evident

when considering the slower convergence and delayed stability observed in YOLOv8x and YOLOv8l models. Despite fluctuations, these YOLOv8l ultimately achieve a level of stability, especially concerning F1-score and mAP@50.

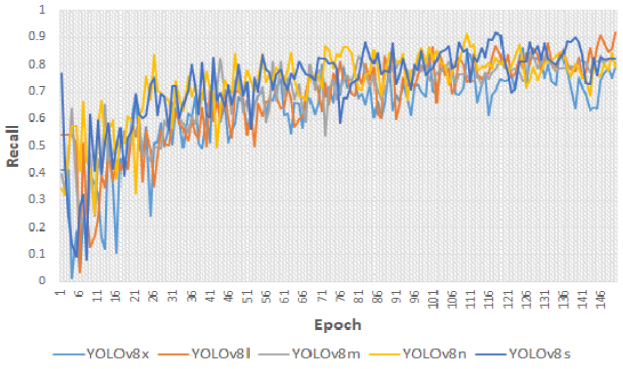

Fig. 5. Evolution of recall for all YOLOv8 variants.

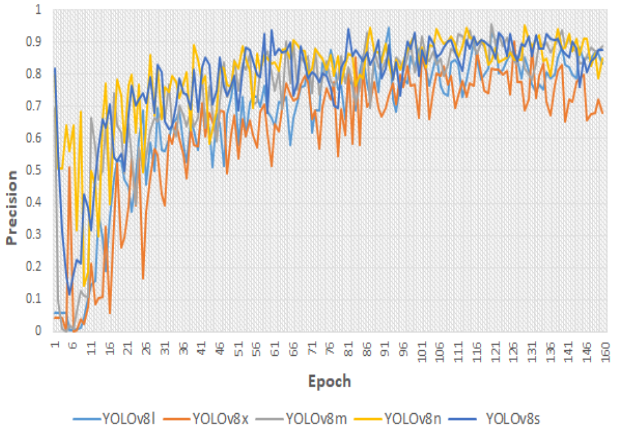

Fig. 6. Evolution of precision for all YOLOv8 variants.

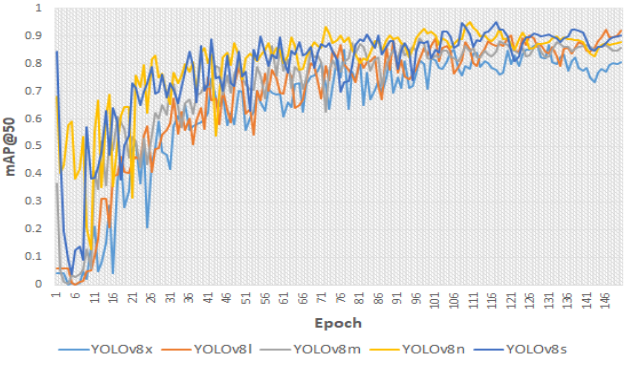

Fig. 7. Evolution of mAP for all YOLOv8 variants.

In the ongoing analysis, table 4 we examine the highest recall achieved along with the corresponding epoch for each variant of the YOLOv8 model during the validation process. The YOLOv8n model achieves a commendable recall of 0.91. This indicates swift learning and robust detection capabilities, even with its relatively uncomplicated structure. This rapid convergence implies its suitability for scenarios with limited computational resources and time constraints, without significant compromises in recall.

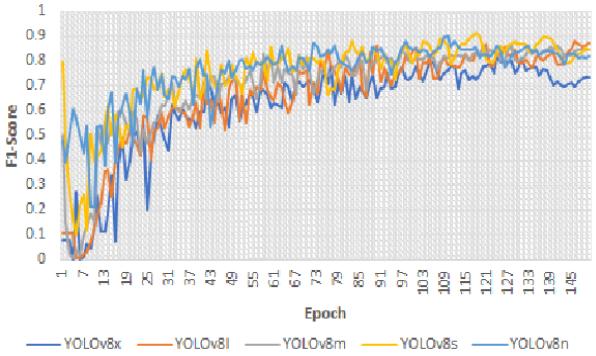

Fig. 8. Evolution of F1 Score for all YOLOv8 variants.

On the other hand, YOLOv8s, despite its increased complexity compared to YOLOv8n, attains recall of 0.91. This indicates that although more training epochs are required, the increased complexity of the model does not lower recall. When considering YOLOv8m, it attains a recall score of 0.848, after 128 epochs. However, it doesn't offer a substantial enhancement when compared to YOLOv8s or YOLOv8n. This prompts inquiries regarding the effectiveness of introducing additional complexity in this variant, as it fails to translate into superior performance.

Interestingly, YOLOv8x manages to attain a recall score of 0.861 by the 126th epoch. This suggests that despite YOLOv8l having a higher level of intricacy compared to YOLOv8s, this increased complexity does not necessarily lead to enhanced performance or faster learning.

At last, YOLOv8l, the sophisticated model, attains an impressive recall rate of 0.913 in epoch 126, surpassing all previous versions in terms of recall performance. What's intriguing is that, despite its increased complexity, it reaches its peak performance more quickly when compared to both YOLOv8s and YOLOv8m.

This implies that YOLOv8l may strike the optimal equilibrium between complexity and performance within this group of models, offering superior detection accuracy without a substantial increase in training time. These findings emphasize that there isn't a straight- forward connection between recall performance and model complexity. While the advanced model, YOLOv8l, attains the highest recall rate, it's clear that not every increase in complexity results in improved performance, as demonstrated by YOLOv8m and YOLOv8x.

Images results of different models is shown in fig 9. according to the figures it is clear that YOLOv8l, YOLOv8x has quite decent detection.

Fig. 9. Here are the image results for various models: (a) The original set of images. (b) Images generated by YOLOv8x.(c) Images produced by YOLOv8l. (d) Images from YOLOv8m. (e) Images generated using YOLOv8s. (f) Images created with YOLOv8n.

4. CONCLUSION

Detecting elderly fall behavior and safeguarding human health are of utmost importance. In this study, an YOLOv8 based network is presented for the purpose of detecting elderly fall. Using custom dataset, the experimental findings

show that YOLOv8l outperforms the other models in recall. but YOLOv8s gives better mAP value. YOLOv8s demonstrated a slight precision advantage over YOLOv8m and YOLOv8l. The model results indicate that YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x achieved mean average precision (mAP@50) values of 0.952, 0.953, 0.882, 0.95, and 0.869, respectively. YOLOv8l outperformed other model with mAP@50 of 95% and best recall of 0.913, respectively. In conclusion, when assessing various architectures and iterations for fall detection, it becomes evident that there isn't a single "optimal" model that performs exceptionally well in every aspect.

The selection of a model should hinge on the particular requirements of the application, taking into account factors such as accuracy, recall, inference time, and the trade-offs among these metrics. Building upon these discoveries, researchers now have a solid groundwork on which to further explore, suggest ground breaking enhancements, and extend the frontiers of this domain. This progress ultimately paves the way for the development of elderly fall detection systems that are both more efficient and effective.

REFERENCES

[1] Y. Chen, Z. Liu, Y. Huang et al., "The aging trend of Chinese population and the prediction of aging population in 2015-2050," Chinese Journal of Social Medicine, vol. 35, no. 5, pp. 480–483, 2018.

[2] A. Joseph, D. Kumar, and M. A. Bagavandas, "Review of epidemiology of fall among elderly in India," Indian J Community Med., vol. 44, no. 2, pp. 166–168, 2019.

[3] B. K. S. Chan, L. M. Marshall, K. M. Winters, K. A. Faulkner, A. V. Schwartz and E. S. Orwoll., "Incident fall risk and physical activity and physical performance among older men," American Journal of Epidemiology, vol. 165, no. 6, December 2006.

[4] K. Wang, G. Zhan, and W. Chen, "A new approach for IoT-based fall detection system using commodity MMWAVE sensors," in Proceedings of the 2019 7th Inter- national Conference on Information Technology: IoT and Smart City, 2019, pp. 197–201.

[5] L. Ren and Y. Peng, "Research of fall detection and fall prevention technologies: A systematic review," IEEE Access, vol. 7, pp. 77 702–77 722, 2019.

[6] K. Wang, G. Zhan, and W. Chen, "A new approach for IoT-based fall detection system using commodity MMWAVE sensors," in Proceedings of the 2019 7th Inter- national Conference on Information Technology: IoT and Smart City, 2019, pp. 197–201

[7] P. V. Er and K. K. Tan, "Wearable solution for robust fall detection," in Assistive Technology for the Elderly. Elsevier, 2020, pp. 81–105

[8] Z. Sheng-lan, Y. Yi-fan, G. Li-fu, and W. Diao, "Re- search and design of a fall detection system based on multi-axis sensor," in Proceedings of the 4th International Conference on Intelligent Information Processing, 2019, pp. 303–309

[9] P. V. Er and K. K. Tan, "Wearable solution for robust fall detection," in Assistive Technology for the Elderly. Elsevier, 2020, pp. 81–105

[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788

[11] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," Advances in neural information processing systems, vol. 28, 2015

[12] R. Girshick, "Fast R-CNN," in Proceedings of the IEEE international conference on computer vision, 2015, pp. 1440–1448.

[13] L. Ma, M. Liu, N. Wang, L. Wang, Y. Yang, and H. Wang, "Room-level fall detection based on ultra-wideband (UWB) monostatic radar and convolutional long short-term memory (LSTM)," Sensors, vol. 20, no. 4, p. 1105, 2020

[14] J. Zhang, C. Wu, and Y. Wang, "Human fall detection based on body posture spatiotemporal evolution," Sensors, vol. 20, no. 3, p. 946, 2020.

[15] Y. Zhu, Y. Zhang, and S. Li, "Fall detection algorithm based on deep vision sensor and convolutional neural network," Opt. Technique, vol. 47, no. 1, pp. 56–61, 2021.

[16] J. R. Cao, J. J. Lu, and X. Y. Wu, "Fall detection algorithm combining motion features and deep learning," Comput. Appl., vol. 41, no. 2, pp. 583–589, 2021.

[17] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," arXiv preprint arXiv:1804.02767, 2018

[18] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," arXiv preprint arXiv:2004.10934, 2020

[19] U. Nepal and H. Eslamiat, "Comparing yolov3, yolov4 and yolov5 for autonomous landing spot detection in faulty UAVS," Sensors, vol. 22, no. 2, p. 464, 2022.

[20] C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng, W. Nie et al., "Yolov6: A single-stage object detection framework for industrial applications," arXiv preprint arXiv:2209.02976, 2022.

[21] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of- freebies sets new state-of-the- art for real-time object detectors," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 7464–7475.

[22] R. Bawankule, V. Gaikwad, I. Kulkarni, S. Kulkarni, A. Jadhav, and N. Ranjan, "Visual detection of waste using yolov8," in 2023 International Conference on Sustainable Computing and Smart Systems (ICSCSS). IEEE, 2023, pp. 869–873

[23] Brief summary of YOLOv8 model structure. Access: "https://github.com/ultralytics/ultralytics/issues/189."

[24] X. Li, W. Wang, L. Wu, S. Chen, X. Hu, J. Li, J. Tang, and J. Yang, "Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection," Advances in Neural Information Processing Systems, vol. 33, pp. 21 002–21 012, 2020.

[25] Uoc, N. Q., Duong, N. T., Son, L. A., and Thanh, B. D. "A Novel Automatic Detecting System for Cucumber Disease Based on the Convolution Neural Network Algorithm." GMSARN International Journal 16 (2022): 295-301.

[26] Verma, A., Singh, A., Bihari, A., Tripathi, S., Agrawal, S., Pandey, S. K., and Verma, S. "Identification of Hate Speech on Social Media using LSTM." GMSARN International Journal 17 (2023): 468-474.