IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.102 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2024

DOI: 10.17148/IJARCCE.2024.13549

Real-Time Human Fall Detection using YOLOv8 and OpenCV

M Rama Bai[1], J Sreedevi[2], Abhijith Korukonda[3], Iragavarapu Sri Rama Saketh[4]

PhD, Department of Emerging Technologies, Mahatma Gandhi Institute of Technology, Hyderabad, India[1]

MTech, Department of Emerging Technologies, Mahatma Gandhi Institute of Technology, Hyderabad, India[2]

BTech, Department of Emerging Technologies, Mahatma Gandhi Institute of Technology, Hyderabad, India[3]

BTech, Department of Emerging Technology, Mahatma Gandhi Institute of Technology, Hyderabad, India[4]

Abstract: Falls among individuals pose serious risks to their health and independence, underscoring the importance of effective fall detection solutions. This study aims to address this critical issue by proposing a novel approach that integrates Computer Vision and Deep Learning for Real-time fall detection and assistance.

Traditionally, fall detection systems have relied on wearable sensors, which, despite their widespread use, often suffer from drawbacks such as false alarms and discomfort for the wearer. In response to these limitations, this project introduces an efficient solution by leveraging Computer Vision and Deep Learning.

The core of this innovative system lies in the integration of the YOLOv8 (You Only Look Once) which is a cutting-edge, real-time object detection algorithm that uses Convolutional Neural Network (CNN) to predict the bounding boxes and class probabilities of objects in input images with Computer Vision. YOLOv8, a variant of the YOLO object algorithm series, has demonstrated superior performance in identifying various objects, and therefore has been used in detecting fall events, with remarkable accuracy and efficiency.

By combining the strengths of YOLOv8 and Computer Vision, this solution offers improved accuracy and reliability in identifying fall events and also enhances the overall user experience by providing timely assistance and ensuring the safety and well-being of individuals.

Keywords: Computer Vision, Deep Learning, Microcontrollers, bounding boxes

I. INTRODUCTION

Falls pose a significant challenge to many people in healthcare and beyond, causing serious injuries and health problems. They are the leading cause of injuries worldwide and the second most common cause of fatal accidents. Seniors are particularly at risk, and the leading cause of brain damage in the elderly is falls. Falling is very dangerous; Approximately 30% of people experience a fall at least once a year, and this rate is likely to increase [1]. The urgency of this problem is clear: In the United States alone, every 11 seconds a person falls and requires emergency room treatment, and every 19 minutes an elderly person dies from a fall [1]. Hospitalizations and deaths from falls are expected to increase; It is estimated that seven adults will die every hour in the United States by 2030 [1]. This anxiety reflects the immediate need to fall and is helpful in falling to prevent further problems.

This research aims to address the need for an advanced fall system that can quickly detect falls and sound the alarm in a timely manner. In medical facilities such as hospitals, clinics, and medical facilities, fall detection systems can improve patient safety and enable rapid response to emergency treatment. Similarly, businesses can improve workplace safety by using fall detection equipment in environments such as factories, construction sites and factories to reduce accidents and reduce workload. In remote and dangerous areas, fall detection systems play an important role in keeping workers safe and speeding rescue operations [2]. In addition, people receiving home care, regardless of their age, can benefit from care services thanks to this device, and caregivers and family members can be seen better. The outcome can be greatly reduced, thus improving the quality of life. This article aims to explore the development, use and potential impact of fall detection in various fields and highlight its importance in improving the safety and consumption health of various ethnic groups [3].

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.102 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2024

DOI: 10.17148/IJARCCE.2024.13549

II. RELATED WORK

The existing methods for fall event detection encompass various approaches utilizing machine learning and deep learning techniques. These methods aim to enhance accuracy and robustness in detecting falls while addressing practical challenges and constraints associated with deployment.

Studies [1], [2], [3], [5], [7], and [8] investigate machine learning and deep learning approaches for fall detection. They emphasize improved accuracy achieved through techniques such as K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Convolutional Neural Network (CNN), and Custom CNN. However, they commonly acknowledge drawbacks like dependencies on wearable devices such as accelerometers and challenges related to data preprocessing and computational complexity.

On the other hand, studies [4], [6], [9], [10], and [11] explore various aspects such as the integration of domain knowledge with machine learning, systematic reviews of machine learning techniques, posture analysis using deep learning, and vision-based fall detection systems. They highlight the effectiveness of machine learning and deep learning in fall detection but also acknowledge challenges like high computational requirements, dataset size limitations, and dependency on video quality for accurate detection.

By synthesizing domain knowledge with machine learning techniques, [4] aims to address complexity and computational demands inherent in fall detection scenarios. This aligns with the emphasis on optimizing computational efficiency in [10] to ensure practical feasibility in real-world applications. Additionally, [11] contributes to non-wearable solutions for fall detection, emphasizing the importance of clear and reliable video footage.

Overall, these studies collectively underscore the advancements and challenges in fall detection methods, providing insights into improving accuracy, robustness, and practical feasibility in real-world scenarios.

III. PROPOSED STYLE

The proposed design methodology integrates the YOLOv8 algorithm with Computer Vision for real-time monitoring of individuals to detect falls. This approach aims to address the challenges associated with traditional fall detection systems, which often rely on wearable sensors and are prone to high rates of false alarms, causing discomfort for users. Instead, this system introduces an approach that leverages computer vision and YOLOv8 to achieve accurate and reliable fall detection while prioritizing user comfort and convenience. Figure 1 provides an overview of the design architecture.

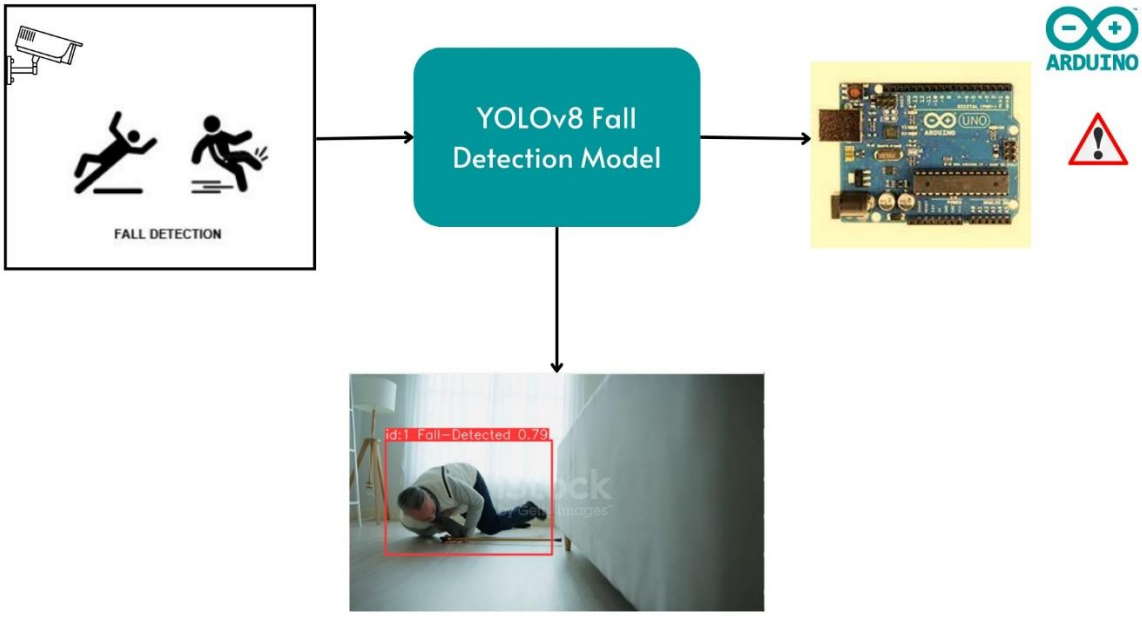

Figure 1: Architecture of Real Time Human Fall Detection System

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.102 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2024

DOI: 10.17148/IJARCCE.2024.13549

A. Data Collection and Augmentation

In the preprocessing phase, the dataset was carefully tailored for fall detection to ensure adaptability and robustness of the model. All images were resized to a standardized format of 1000x800 pixels, maintaining white edges when necessary to create a visually cohesive representation and promote consistency and efficient processing by the model. Augmentations played a crucial role in diversifying the dataset, exposing the model to a broad range of challenging scenarios. Each training example produced two outputs, contributing to a varied and comprehensive learning experience. Augmentations included rotation, introducing random angles between -5° and +5° to simulate varying orientations of falling instances, as well as horizontal and vertical shear effects ±10° in both directions to deform parts of the images and add complexity representative of dynamic fall scenarios. To adapt to variations in color or its absence, grayscale was applied to 10% of the images, and adjustments to hue, saturation, brightness, and exposure were made within specified ranges to challenge the model under different environmental settings. Color-related augmentations enhance the model's resilience to real-world conditions. Simulating scenarios with reduced clarity, blur was applied up to 2.5 pixels, and noise was introduced to 1.8% of pixels, adding randomness to the dataset. Bounding box augmentations were specifically tailored to instances with localized objects, ensuring the model's capability to handle varying scales and perspectives. These augmentations included crop, rotation, shear, brightness, exposure adjustments, blur, and noise, contributing to creating a diverse and challenging dataset. This meticulous approach to preprocessing and augmentations, applied to a dataset of 8,500 images with a distribution of 7,000 for training and 1,500 for validation, ensures that the fall detection model is well-equipped to handle a wide spectrum of real-world scenarios, enhancing its adaptability and generalization capabilities in alignment with the goal of creating a robust and effective fall detection system.

A. crucial component of the project is the data.yaml file. This YAML file is meticulously crafted to encapsulate essential information necessary for seamless interaction between the curated fall detection dataset and the YOLOv8 model during both training and validation stages. The 'data.yaml' file serves as a comprehensive configuration document, specifying key elements that govern the training process. Within this file, the class names are defined, with a singular class named "Fall-Detected" indicating the presence of a fall event. The 'nc' parameter is set to 2, aligning with the focus on detecting falls as the primary class. Furthermore, the file outlines the file paths for the training and validation sets, providing the necessary directory references for the YOLOv8 model to access and utilize the images during different phases of model development. The 'data.yaml' file plays a pivotal role in orchestrating a harmonious synergy between the curated fall detection dataset and the YOLOv8 model, ensuring precise and accurate training and validation.

B. Training the YOLOv8 Model with Custom Dataset

In the pursuit of developing a highly effective fall detection model, a meticulous approach was taken to establish an optimal training environment within the Jupyter Notebook IDE. Leveraging GPU acceleration through CUDA, an environment capable of efficiently handling the complexities of training large-scale models was created. To streamline the training procedures further, the Ultralytics package, a versatile toolkit renowned for its extensive support in computer vision tasks, was integrated. This package played a pivotal role in seamlessly incorporating YOLOv8 into the workflow, providing not only a robust implementation of the YOLO model but also access to a pretrained model, serving as a strong foundation for the fall detection system.

The training process began with the importation of the YOLO model, specifically opting for the yolov8s architecture from the Ultralytics package. This model, combined with the meticulously collected fall dataset and the corresponding ‘data.yaml’ configuration file, formed the essential inputs for the training endeavors. To optimize training parameters, experiments were conducted with different numbers of epochs, including 10, 15, 20, 25, and 30. An observed improvement in accuracy was noted until the 25th epoch, after which there wasn't significant improvement up to the 30th epoch. Therefore, the decision was made to proceed with 25 epochs for training. Spanning over these epochs, the training parameters were further optimized by setting the image size (imgsz) to 800, leveraging the computational efficiency of GPU processing for accelerated convergence.

Throughout the training phase, the model.yaml configuration dynamically adjusted to accommodate the singular class, “Fall-Detected,” signaling a departure from the default 80 classes to align with the specific use case. The resulting model summary unveiled an intricate architecture with 225 layers, 11,135,987 parameters, and 28.6 GFLOPs, underscoring its complexity and proficiency in nuanced fall detection.

The culmination of the training phase yielded a crucial artifact – the .pt file, encapsulating the learned parameters of the model. This file served as the linchpin for the real-time monitoring of falls, enabling the practical implementation of the fall detection system in real-world scenarios.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.102 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2024

DOI: 10.17148/IJARCCE.2024.13549

C. Evaluation

A pivotal step in the design and methodology involved a meticulous evaluation of key performance metrics to assess the effectiveness of the custom YOLOv8 model in elderly fall detection. Leveraging the confusion matrix, a comprehensive analysis encompassed metrics such as accuracy, false positives, false negatives, precision, recall, and the F1-score.

This metric evaluation serves as a crucial checkpoint in understanding the model's behavior and identifying areas for potential refinement. This analysis of metrics lays the foundation for a thorough assessment of the model's performance in real-world scenarios. The insights gleaned from this evaluation will inform further iterations and enhancements to ensure the robustness and reliability of the fall detection system.

D. Real Time Monitoring with OpenCV

To deploy the fall detection model in real-world scenarios, the researchers seamlessly integrated the OpenCV library, a versatile computer vision toolkit renowned for its extensive capabilities. OpenCV provided a robust platform for implementing the YOLOv8 model on both test videos and live camera feeds, enabling real-time monitoring of fall events.

The integration began by importing the YOLO class from the Ultralytics package and initializing the model using the trained weights from the "my\textunderscore model.pt file". Leveraging the power of OpenCV, video frames were accessed either from a specified video file or through a connected camera, allowing for flexible switching between pre-recorded videos and live monitoring. The real-time monitoring script utilized the YOLOv8 tracking capabilities provided by Ultralytics. For each frame in the video stream, the model performed real-time object tracking with a confidence threshold set at 0.5, persisting tracks between frames for enhanced continuity.

The results, enriched with tracking information, were overlaid onto the original frame using OpenCV's visualization functions. The annotated frame, depicting the detected fall instances and their tracking details, was then displayed in a dedicated window named "YOLOv8 Tracking."

A notable feature of the implementation was user interactivity. The script allowed for the termination of the monitoring loop by pressing 'q,' ensuring a seamless and user-friendly experience. Moreover, the integration of OpenCV facilitated adaptability, enabling easy deployment on various platforms and devices.

This real-time monitoring script not only showcased the effectiveness of the fall detection model but also demonstrated its applicability in live scenarios, reinforcing its potential for deployment in real-world environments. The combination of YOLOv8 and OpenCV provided a powerful solution for real-time fall detection, contributing to the overarching goal of enhancing the safety and well-being of individuals.

E. Integration with Arduino and Alert System

During the integration phase of the design, the fall detection system was seamlessly merged with hardware components to enhance its real-world applicability. The selected hardware solution for this integration was Arduino, a versatile microcontroller platform. The objective was to connect the fall detection system to a sound alert generator, creating a responsive alert mechanism for immediate intervention upon detecting a fall in real-time.

Arduino Integration:

The Arduino microcontroller functioned as the central component for interfacing between the fall detection system and the alert system. The integration involved establishing a communication link between the YOLOv8 model, operating on the primary system, and the Arduino board. This communication enabled the seamless transmission of fall detection signals from the software to the hardware components.

Alert System:

The hardware setup consisted of a sound alert generator configured to activate upon detecting a fall. The fall detection system continuously monitored the real-time camera feed, and upon identifying a fall event, a trigger signal was sent to the Arduino board. In response, the Arduino activated the sound alert generator and displayed text on a 16x2 LCD Display, providing an immediate and audible notification to alert caregivers or relevant personnel. Figure 2 gives the Block diagram of the system.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.102 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2024

DOI: 10.17148/IJARCCE.2024.13549

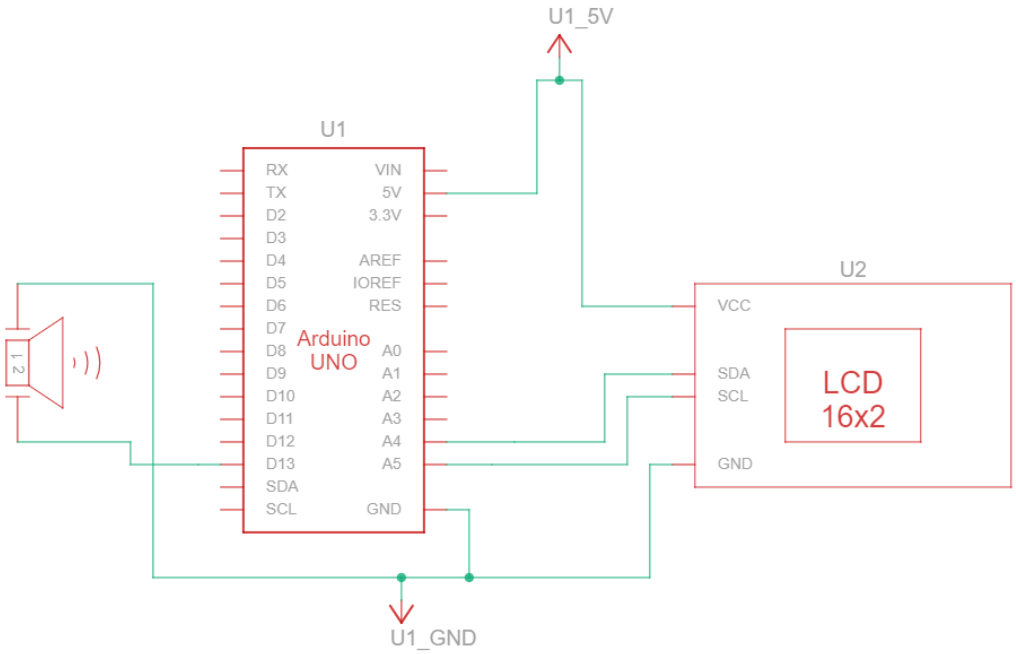

Figure 2: Block Diagram of the Human Fall Detection System

This integration of software and hardware components represents a critical step in enhancing the practical utility of our fall detection system. By connecting the system to a responsive alert mechanism, we ensure that timely assistance is provided in the event of a fall, thereby contributing to the overall safety and well-being of individuals under monitoring. The successful amalgamation of computer vision with hardware components underscores the versatility and effectiveness of our integrated fall detection solution.

IV. EXPERIMENTAL RESULTS

In this critical phase of the evaluation, a comprehensive assessment of the custom-trained YOLOv8 model was conducted using a validation set. Leveraging Ultralytics YOLOv8.1.18 and Python-3.10.12 with CUDA support on a Tesla T4 GPU, the model demonstrated exceptional computational efficiency, featuring 168 layers and a total of 11,125,971 parameters. The integrated architecture of the model indicated optimization for streamlined inference.

During the validation process, the model proficiently scanned through 899 images in the designated validation dataset. Impressively, it successfully identified and processed each image without encountering any background elements or corrupt data. The subsequent analysis focused on class-specific metrics, particularly emphasizing the "Fall Detected" class. The model exhibited a precision of 0.8, a recall of 0.806, and a mean Average Precision at IoU 0.5 (mAP50) of 0.86. Additionally, it demonstrated a competitive mean Average Precision from IoU 0.5 to 0.95 (mAP50-95) of 0.499.

The model's performance speed per image during detection further underscored its efficiency. With 1.2ms allocated to preprocessing, 13.5ms to inference, and 2.8ms to post-processing, the model displayed remarkable speed in processing fall instances. These results affirm the effectiveness of the fall detection system, demonstrating its capability to deliver accurate and swift detection, crucial for real-time monitoring and timely intervention in critical situations.

The model achieved an overall accuracy of 80.2%, correctly classifying 802 out of 1000 instances. Out of the 198 instances classified as “fall detected”, 197 were correct, resulting in a precision of 97%. This indicates that 97% of the time the model predicted a fall, it was correct.

Regarding the actual falls, the model correctly classified 600 out of 800 instances, achieving a recall of 80%. This means that the model missed 200 out of 800 actual falls.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.102 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2024

DOI: 10.17148/IJARCCE.2024.13549

Furthermore, the model accurately classified all 200 background instances, achieving a specificity of 100%. This implies that the model never incorrectly classified a background instance as a fall.

The F1-score, which is a harmonic mean of precision and recall, serves as a good measure of the overall performance of the model. In this case, the F1-score is 88%, indicating good performance.

Overall, the confusion matrix demonstrates that the fall detection model performs well. It exhibits high accuracy, precision, and F1-score, and it effectively identifies falls. However, it does miss some falls, highlighting a potential limitation that needs to be considered.

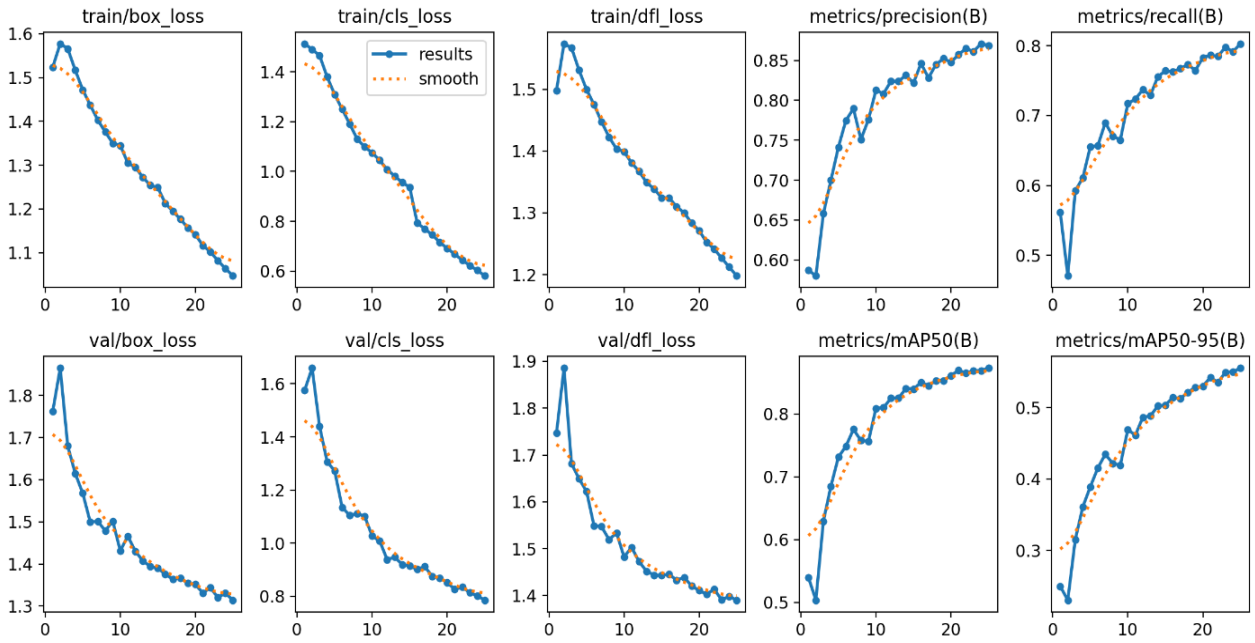

Figure 3: Graphs representing change in Loss and Precision with epochs

The Figure 3 illustrates graphs representing different graphs that visualize the performance of a model during training. The x-axis typically represents the number of training epochs or iterations, and the y-axis represents the value. A lower loss value generally indicates better model performance. We can observe that the loss value decreases over the training and the precision and recall values have increased over the training.

V. CONCLUSION

In conclusion, the integration of computer vision capabilities with Arduino microcontroller functionality has resulted in an effective fall detection system. The evaluation of the custom-trained YOLOv8 model has demonstrated remarkable computational efficiency and accuracy in real-time fall detection. Through the validation process, the YOLOv8 model has shown robustness, achieving a precision of 0.8 and a recall of 0.806 in identifying fall events. With a mean Average Precision at IoU 0.5 (mAP50) of 0.86 and a competitive mean Average Precision from IoU 0.5 to 0.95 (mAP50-95) of 0.499, the model consistently distinguishes fall instances from background elements.

The integration of Arduino microcontrollers has enhanced the system's functionality, providing a versatile and cost-effective platform for real-time monitoring and response mechanisms. Arduino boards act as central processing units, receiving input from cameras, analyzing data using the YOLOv8 algorithm, and triggering appropriate actions such as activating LED indicators and sounding alarms to signal a fall event.

The fall detection system exhibits impressive efficiency with rapid processing speed, requiring only 17.5ms per image for preprocessing, inference, and post-processing. This efficiency facilitates timely intervention, critical for ensuring the safety and well-being of elderly individuals.

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.102 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 5, May 2024

DOI: 10.17148/IJARCCE.2024.13549

Despite these achievements, several minor drawbacks were identified that warrant further improvement. For instance, performance in low-light environments could be enhanced through advanced image preprocessing techniques or by integrating infrared sensors for improved visibility. Additionally, exploring alternative alert mechanisms such as vibrating devices could mitigate the limited effectiveness of sound-based alerts in noisy environments.

Overall, the system represents a significant advancement in fall detection technology, promising improved accuracy, reliability, and user experience. Future endeavors will focus on refining and optimizing the system for real-world deployment, ultimately enhancing the quality of life for individuals and caregivers.

REFERENCES

[1]. B. McMahan et al., "Communication-Efficient Learning of Deep Networks From Decentralized Data", *Artificial Intelligence and Statistics Proc. PMLR*, vol. 10, no. 1, pp. 1273-82, 2017.

[2]. C. En Guo, S.-C. Zhu and Y. N. Wu, "Primal Sketch: Integrating Structure and Texture", *Computer Vision and Image Understanding*, vol. 106, no. 1, pp. 5-19, 2007.

[3]. S. Zhang, C. Zhu, J. K. O. Sin, and P. K. T. Mok, "A novel ultrathin elevated channel low-temperature poly-Si TFT," *IEEE Electron Device Lett*., vol. 20, no. 2, pp. 569–571,1999.

[4]. Hogade, N., Pasricha, S. and Siegel, H.J, "Energy and Network Aware Workload Management for Geographically Distributed Data Centers". *IEEE Transactions on Sustainable Computing*, vol.7, no. 2, pp.400–413. 2021

[5]. A. Wierman, Z. Liu, I. Liu and H. Mohsenian-Rad, "Opportunities and challenges for data center demand response", *Proc. Int. Green Comput. Conf.*, vol.7, no. 6, pp.1-10, 2014.

[6]. J. D. Jenkins et al., "The benefits of nuclear flexibility in power system operations with renewable energy", *Appl. Energy*, vol. 22 no. 2, pp. 872-884, 2018.

[7]. Haoying Dai, Yanne Kouomou Chembo, "RF Fingerprinting Based on Reservoir Computing Using Narrowband Optoelectronic Oscillators", *Journal of Lightwave Technology*, vol.40, no.21, pp.7060-7071, 2022.

[8]. Floris Van den Abeele, Jeroen Hoebeke, Girum Ketema Teklemariam, Ingrid Moerman, Piet Demeester, "Sensor Function Virtualization to Support Distributed Intelligence in the Internet of Things", *Wireless Personal Communications*, vol.81, no.4, pp.14-18, 2015.

[9]. J. Hwang, J. Kim and H. Choi, "A review of magnetic actuation systems and magnetically actuated guidewire-and catheter-based microrobots for vascular interventions", *Intell. Serv. Robot*., vol. 13, no. 1, pp. 1-14, 2020.

[10]. D. G. Feitelson, D. Tsafrir and D. Krakov, "Experience with using the parallel workloads archive", *J. Parallel Distrib. Comput*., vol. 74, no.3, pp. 2967-2982, 2014.

[11]. B. Accou, J. Vanthornhout, H. V. Hamme and T. Francart, "Decoding of the speech envelope from eeg using the vlaai deep neural network", *Scientific Reports*, vol. 13, no. 1, pp. 812, 2023.

[12]. Serim Lee, Nahyun Kim, Junhyoung Kwon, Gunhee Jang, "Identification of the Position of a Tethered Delivery Catheter to Retrieve an Untethered Magnetic Robot in a Vascular Environment", *Micromachines*, vol.14, no.4, pp.724, 2023.