International Journal of Information Security, Privacy and Digital Forensics Vol. 9, No. 1, June, 2026

An Edge-Based Internet of Things (IoT) Architecture for Privacy-Preserving Fall Detection in Elderly Care

Ogbu Nwani Henry*[1]
Department of Computer Science, Ebonyi State University, Ebonyi State, Nigeria.
ogbu.henry@ebsu.edu.ng

Ifeanyi S. Nwoye[2]
Department of Computer Science, Ebonyi State University, Ebonyi State, Nigeria
sinless4221@gmail.com

Ani Maxwell A[3]
Department of Computer Science, Ebonyi State University, Ebonyi State, Nigeria
ogbumaxwell67@gmal.com

Anyim Chukwuemeka[3]
Department of Computer Science, David Umahi Federal University of Health Sciences, Uburu, Ebonyi State.
anyimemy@gmail.com

ABSTRACT

Falls are a leading cause of injury and death among older adults. Most fall detectors rely on wearable sensors that send data to the cloud, which adds delay, exposes sensitive movement data and stops working when the internet is unavailable. This paper presents an edge-based IoT system that processes all sensor data locally. An MPU6050 motion sensor connected to a NodeMCU ESP8266 microcontroller streams data over Wi-Fi to a local Flask server running a hybrid CNN-LSTM model that separates falls from normal activities. On live data from several participants the system reached 97.4% accuracy. End-to-end alert latency was 1.2 s, compared with 4.3 s for the same model deployed in the cloud. Raw motion data never leaves the local network, notifications are encrypted before they are sent, and the system keeps working during internet outages. Average current draw is about 120 mA, which is comparable to commercial wearables. The results show that edge processing can match cloud accuracy while giving faster response and stronger privacy, making it a practical option for fall detection in elderly care

Keywords: Edge Computing; Internet of Things; Fall Detection; Privacy-Preserving; Hybrid CNN-LSTM Model; Elderly Care

1. INTRODUCTION

Falls are a major cause of injury and death among older adults. The World Health Organization (2023) estimates that about 37.3 million fall-related incidents each year need medical attention. Between 28% and 35% of people aged 65 and above fall at least once a year, rising to 32–42% for those aged 70 and above (Villa & Casilari, 2025; Kumar, Gupta, & Rawat, 2023), largely because balance, muscle strength and reflexes decline with age (Singh & Gupta, 2024). These figures have driven interest in automated fall detection.

Early systems used cameras and depth sensors. They performed well in the laboratory but raised privacy concerns, were sensitive to lighting and were complicated to install (Patel & Ross, 2023). Wearable accelerometers and gyroscopes offered a non-invasive alternative and have exceeded 90% accuracy in laboratory studies (Chen, Wang, & Yu, 2022), but sensor placement, movement variability and false alarms still limit real-world performance (Yasmin et al., 2024).

Internet of Things (IoT) connectivity enabled remote monitoring, with data sent to cloud platforms for analysis and alerts forwarded to caregivers (Qadir et al., 2023; Karthikeyan, Ramesh, & Anand, 2022). Cloud designs have drawbacks, however. Transmission delay can range from seconds to minutes (Mao et al., 2024), and typical cloud systems need 3–5 s from fall to alert. Continuous uploading drains batteries. Sending sensitive physiological data over the internet makes older users reluctant to accept these systems (Patel & Ross, 2023; Irshad & Alzahrani, 2025), and the whole system stops when the connection fails.

This paper takes a different approach. All computation is done locally on an edge gateway, and raw sensor data never leaves the local network. This lowers latency and bandwidth use, preserves privacy by design and allows operation during internet outages. Advances in TinyML and edge computing show that capable models can run efficiently on constrained hardware (Cong et al., 2024; Prasad et al., 2024). The contributions of this work are: (i) a three-layer edge architecture for privacy-preserving fall detection; (ii) a privacy-by-design approach that avoids

external transmission of motion data; (iii) experimental results of 97.4% accuracy and 1.2 s alert latency; (iv) a controlled edge-versus-cloud latency comparison using the same model; and (v) a demonstration of operation during internet outages. Section 2 reviews related work, Section 3 describes the materials and methods, Section 4 presents the results and discussion, and Section 5 concludes.

2. RELATED WORK

Cloud-based systems. Early IoT fall detection was largely cloud-based: wearables sent data to remote servers, models classified it, and alerts were sent back to caregivers. Kumar et al. (2023) used IMUs with a cloud back end and reported about 90% accuracy with Random Forest and SVM classifiers, but an average alert time of 4.2 s is a concern in emergencies, and participants repeatedly raised privacy concerns. Karthikeyan et al. (2022) showed a similar cloud framework was feasible but left network dependence and data security unresolved and recommended edge computing as future work. Salem and Hasan (2024) reached 95% accuracy with a CNN-LSTM hybrid on the SisFall dataset, but evaluated it offline with no real-time IoT validation. Yasmin et al. (2024) compared SmartFall architectures and found the server-based design best for accuracy and the serverless design worst for latency, which illustrates the accuracy–latency trade-off.

Privacy and security. Patel and Ross (2023) found that most researchers concentrate on algorithmic accuracy and overlook the risks of sending sensitive data to third parties, which hinders adoption of wearable fall recognition systems. Comprehensive edge-based designs are few; Ahmed and Mehmood (2023) argued that local inference architectures keep data under user control and should be central to privacy-preserving design. Surveys of IoMT and IoHT security show that authentication, intrusion detection, privacy preservation, real-time monitoring and secure data handling remain open challenges (Shammar et al., 2025; Irshad & Alzahrani, 2025; Kushwaha et al., 2025), with computational complexity, real-time adaptability and energy efficiency as persistent constraints.

Edge computing and TinyML. These approaches move inference closer to the sensors. Qadir et al. (2023) built NodeMCU prototypes and found that edge processing reduces latency and preserves privacy, although models transferred poorly between users. Zhao et al. (2022) cut memory use by 75% with quantisation-aware training while preserving accuracy. Prasad et al. (2024) validated real-time TinyML fall detection on constrained devices. Cong et al. (2024) deployed a 9,623-byte quantised model on an ESP32-S3 that achieved 97.8% accuracy on SisFall and 97.4% on self-collected data, with 14 ms inference time. Villa and Casilari (2025) combined threshold-based detection with CNN-LSTM classification over LoRa, reaching 96.67% sensitivity, 100% specificity and 178 hours of continuous operation. Mao et al. (2024) applied knowledge distillation across a multilayer mobile edge computing framework, improving accuracy by 11.65% and reducing latency by 46.67% on SisFall.

Research gaps. Several gaps remain. Many models are accurate offline but have not been tested in real IoT or edge settings (Salem & Hasan, 2024; Chen et al., 2022). Privacy is rarely a design objective (Patel & Ross, 2023; Shammar et al., 2025). Cloud latency is still too high for emergencies (Kumar et al., 2023; Mao et al., 2024). Studies that compare edge and cloud using the same model and hardware are scarce. Finally, most work addresses either algorithm performance or IoT communication, rarely both together with privacy (Ahmed & Mehmood, 2023; Kushwaha et al., 2025). This work fills these gaps with a complete edge stack, a quantitative edge–cloud comparison, privacy through local processing, energy validation and demonstrated offline operation.

3. MATERIALS AND METHODS

3.1 System Architecture

The proposed system has three layers, shown in Figure 1, which together support real-time private monitoring.

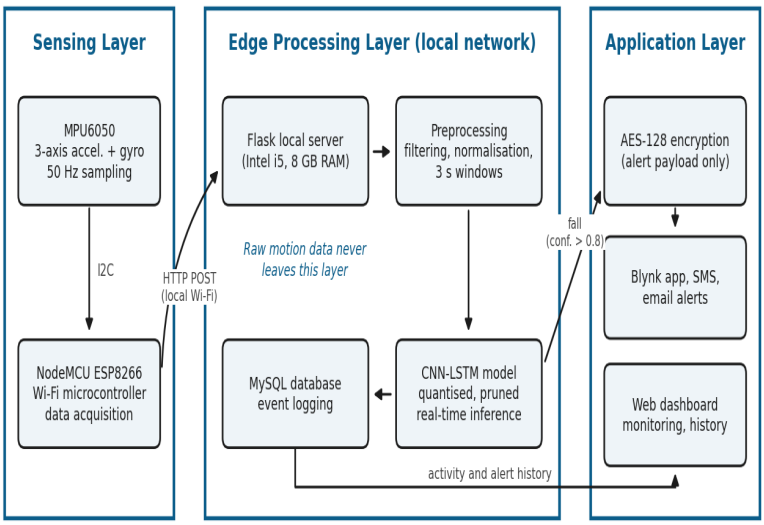


Figure 1: *The edge-based IoT architecture.*

Sensing layer. A wearable module combines an MPU6050 (3-axis accelerometer and 3-axis gyroscope, sampled at 50 Hz) with a NodeMCU ESP8266 that handles sensor set-up, reading and Wi-Fi communication. It is powered by a 5 V rechargeable battery pack.

Edge processing layer. Sensor streams are sent by HTTP POST to a Flask server on a standard computer (Intel i5, 8 GB RAM) on the same local network. The server filters noise, normalises and segments the data, and passes it to the trained CNN-LSTM model for real-time classification. Nothing is transmitted over the internet at this stage.

Application layer. A Flask web dashboard shows a real-time activity graph, alert history and system status. When a fall is detected with high confidence (>0.8), the system notifies caregivers through the Blynk IoT mobile app, SMS and e-mail. All events are logged in a MySQL database for later analysis.

3.2 Hardware Components

The components are summarised in Table 1. The NodeMCU ESP8266 has enough memory for data collection and transmission, the MPU6050 offers programmable full-scale ranges and an I2C interface, and deep-sleep modes extend battery life. The assembled system is shown in Figure 2.

Table 1: Hardware Component Specifications

Component	Key Specifications	Power Consumption	Figures
NodeMCU ESP8266	80MHz clock, 4MB flash, Wi-Fi	70mA active	
MPU6050	16-bit ADC, ±2g to ±16g, ±250°/s to ±2000°/s	3.6mA	
Power Supply	5V/2A Li-ion battery with DC-DC converter	-	
Local Server	Intel i5, 8GB RAM, 500GB, HDD	-	

For collecting and sending the data from the sensors, the NodeMCU ESP8266 has enough memory. The MPU6050 has programmable full scale ranges and I2C communication interface. The battery life is increased with the use of deep sleep modes. The system is assembled as shown in figure 2.

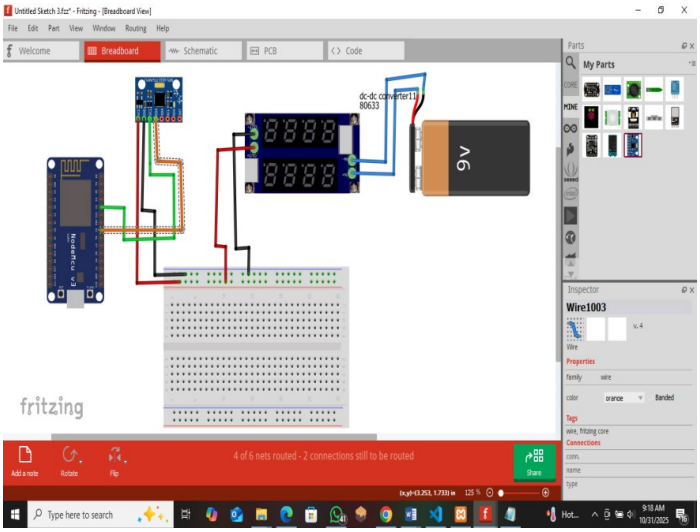


Figure 2: *System hardware component interfacing.*

3.3 Dataset Description

The SisFall benchmark was used for training. It contains accelerometer and gyroscope recordings from 38 subjects performing 19 activities of daily living (ADLs) and 15 types of fall, including walking, sitting, standing, lying and simulated falls (forward, backward and lateral, with and without attempts to recover). It comprises about 4,500 fall events and 3,000 ADL sequences recorded at 50 Hz with 16-bit resolution.

Primary data were also collected with the prototype hardware from five volunteers (aged 22–45) who performed controlled activities and simulated falls on padded mats. This data adapted the model to the sensor characteristics and local conditions. All data were combined into six attributes (tri-axial accelerometer and gyroscope values) and split into 3-second windows (150 samples) with 50% overlap, since falls tend to occur within this period (Chen et al., 2022). The data format is shown in Table 2.

Table 2: *Sample sensor data format.*

accel_x	accel_y	accel_z	gyro_x	gyro_y	gyro_z	activity
0.234	9.812	0.567	0.012	0.008	0.015	standing
0.456	9.801	0.678	0.023	0.011	0.018	walking
3.234	7.456	2.345	0.345	0.567	0.234	falling

3.4 CNN-LSTM Model Architecture

The model combines convolutional neural networks (CNNs), which extract local patterns from motion signals, with Long Short-Term Memory (LSTM) networks, which learn temporal dependencies. This suits falls, which involve abrupt changes in acceleration followed by a defined temporal progression of pre-fall movement, impact and post-fall posture (Salem & Hasan, 2024). The architecture is as follows.

- **Input:** 3-second windows of 150 time steps × 6 features.
- **CNN layers:** Conv1D (64 filters, kernel size 3, ReLU), Conv1D (128 filters, kernel size 3, ReLU) and MaxPooling1D (pool size 2). The convolution is given in Equation (1).

$$y_i = f\left(\sum_{k=1}^{K} w_k x_{i+k-1} + b\right) \quad (1)$$

where y_i is the output feature at position i, w_k the kernel weights, x_{i+k-1} the input sequence, b the bias and f(·) the ReLU activation.

- **LSTM layers:** LSTM (128 units, return sequences) followed by LSTM (64 units). The cell operates through the gates in Equations (2)–(7).

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \quad (2)$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \quad (3)$$

$$\tilde{C}_t = \tanh(W_c[h_{t-1}, x_t] + b_c) \quad (4)$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \quad (5)$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad (6)$$

$$h_t = o_t \odot \tanh(C_t) \quad (7)$$

Here σ is the sigmoid function, ⊙ is element-wise multiplication, h is the hidden state, x the input, and W and b are learnable weights and biases.

- **Dense layers:** Dense (64 neurons, ReLU), Dropout (0.5) to limit over-fitting, and a 3-neuron softmax output for the classes Normal Activity, Fall and Transition (Equation 8).

$$P(y = j \mid x) = \frac{e^{z_j}}{\sum_k e^{z_k}} \quad (8)$$

- **Training:** Adam optimiser (learning rate 0.001), categorical cross-entropy loss (Equation 9), batch size 32, up to 50 epochs and early-stopping patience of 10.

$$L = -\sum_j y_j \log \hat{y}_j \quad (9)$$

3.5 TinyML Optimisation

The trained model was compressed for resource-limited deployment. Quantisation converted weights from 32-bit floating point to 8-bit integers, cutting model size by about 75% without loss of accuracy and speeding up inference. Pruning removed weak connections (weights below 0.01) to simplify the model further. The optimised model is about 2.5 MB and runs on the local Flask server without any cloud dependence.

3.6 Evaluation Metrics

Standard multi-class metrics were used (Table 3), where TP are falls correctly detected, TN non-fall activities correctly identified, FP normal activities wrongly classified as falls, and FN falls the system missed.

Table 3: Classification Performance Metrics

Metric	Mathematical Expression
Accuracy	$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$
Precision	$Precision = \frac{TP}{TP + FP}$
Recall (Sensitivity)	$Recall = \frac{TP}{TP + FN}$
F1-Score	$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$

Where:

TP (True Positives): Correctly detected falls
TN (True Negatives): Correctly identified non-fall activities
FP (False Positives): Normal activities incorrectly classified as falls
FN (False Negatives): Falls that were not detected by the system

Latency Measurement

In addition to the classification accuracy, the responsiveness of the system was assessed with a latency analysis. Latency is the time in seconds that occurs between the fall and the delivery of an alert to the caregiver device. The measure of latency is summarized in Table 4.

Table 4: Latency Measured

Metric	Mathematical Expression	Equation No.	Description
Latency	$Latency = t_{alert} - t_{event}$	(7)	Measures response time between fall event detection and alert notification.

Where:

t_{event} = timestamp of fall occurrence
t_{alert} = timestamp of alert delivery

The latency value was averaged over 100 independent test runs to ensure statistical reliability and consistency.

4. RESULTS AND DISCUSSION

4.1 Model Performance

The CNN-LSTM model classified the test data successfully; results are given in Table 4.

Table 5: Model Performance Metrics

Metric	Value (%)
Accuracy	97.4
Precision	95.8
Recall	96.9
F1-Score	96.3
True Positive Rate	96.9
False Positive Rate	3.1

The 97.4% accuracy is 2.4 percentage points higher than that of the next best comparable system (Salem & Hasan, 2024), which supports the value of the hybrid architecture combined with edge optimisation. The high recall (96.9%) shows that falls were detected reliably. The slightly lower precision (95.8%) is probably due to false alarms during sudden non-fall activities such as jumping or sitting down quickly, and the F1-score of 96.3% shows that precision and recall are well balanced. Figure 3 shows the confusion matrix and Figure 4 a screenshot of the model correctly predicting a fall.

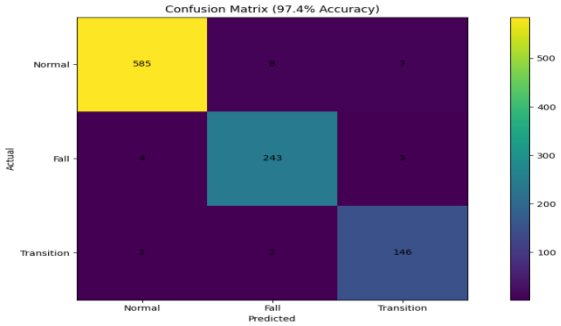


Figure 3: Confusion Matrix for the Classification Results

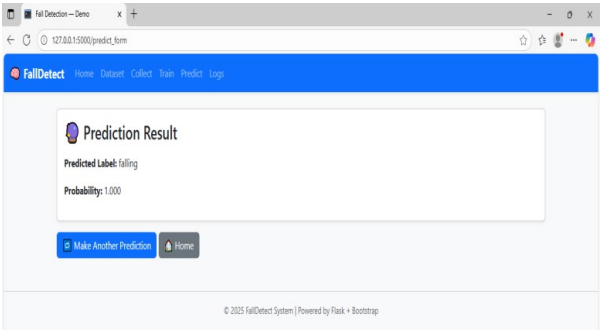


Figure 4: Screen Shot of Model Performance

4.2 Latency Comparison: Edge vs. Cloud

A key contribution of this study is quantitative comparison between edge-based and cloud-based architecture. For cloud comparison, the identical CNN-LSTM model was deployed on a remote server (AWS EC2 instance) with data transmission over internet, while the edge configuration used local processing. Table 3 presents the latency comparison results.

Table 6: Latency Comparison: Edge vs. Cloud Architecture

Component	Edge System (ms)	Cloud System (ms)
Sensor Data Acquisition	50	50
Local Processing	150	150
Data Transmission	200	2,500
Cloud Processing	N/A	800
Alert Delivery	800	800
Total Latency	**1,200**	**4,300**

The 97.4% accuracy is 2.4% higher than the accuracy of the next best comparable system (Salem & Hasan, 2024), which is proof of the effectiveness of hybrid architecture along with edge optimization. The model showed good accuracy in fall detection and exhibited a very high recall (96.9%) which means that the model was able to detect falls reliably. The slightly reduced accuracy (95.8%) may be due to false alarms when performing sudden, but not fall, activities like jumping or quick sitting. The F1 score of 96.3% indicates that the precision and recall were balanced. The confusion matrix of the classification results is shown in figure 3.

This difference is clinically significant in the case of emergency fall detection. This 1.2-second alert allows for quicker caregiver response – around 3 seconds before cloud systems – can help minimize the severity of injuries by intervening earlier. The comparison of latencies is shown graphically in figure 5.

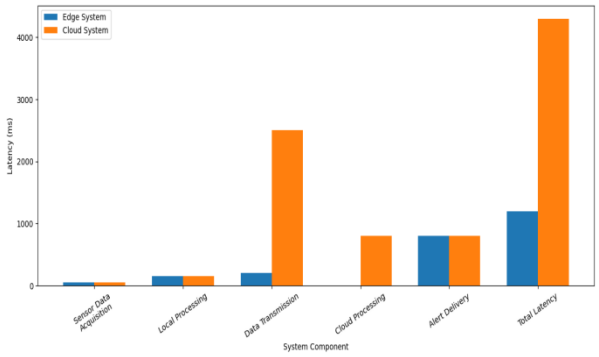


Figure 5: Latency Comparison Graph

4.3 Privacy and Security Analysis

- **Data localisation.** Raw accelerometer and gyroscope data never leave the local network. Only the alert type and timestamp are sent to external services (Blynk, SMS, e-mail), and they are encrypted with AES-128 first. Sensitive movement patterns are therefore never exposed to third-party servers.
- **Internet independence.** Fall detection and the local dashboard continue during internet outages. Alerts are queued and delivered when connectivity returns.

- **Data protection.** The design is consistent with privacy regulations such as the Nigeria Data Protection Regulation and the GDPR, since no personal health data leaves the user's control.

4.4 Power Consumption

The NodeMCU drew about 120 mA on average during sensing and Wi-Fi transmission, and about 15 mA in deep-sleep mode where implemented. With a 2000 mAh battery this gives 16–18 hours of continuous use, which duty cycling can extend. A draw of 120 mA is similar to commercial fitness trackers (100–150 mA), which suggests the device is viable for home use. Edge processing saves power compared with cloud solutions because transfers are short-range and local and there is no constant upload to the internet, although Wi-Fi radio power varies with signal strength.

4.5 Comparison with Existing Systems

Table 6 compares the proposed system with representative studies on key features.

Table7: Comparison with Existing Systems

Feature	**Chen et al. (2022)**	**Kumar et al. (2023)**	**Salem & Hasan (2024)**	**This Work**
Architecture	Wearable + Cloud	Wearable + Cloud	Offline only	Wearable + Edge
Accuracy (%)	92.0	90.0	95.0	97.4
Latency (s)	Not reported	4.2	Not applicable	1.2
Privacy-Preserving	No	No	N/A	Yes
Offline Operation	No	No	N/A	Yes
Real-Time Alerts	Limited	Yes	No	Yes

The proposed system compares favourably on every feature, especially latency and privacy. The hybrid CNN-LSTM model and the combination of benchmark and primary data may have contributed to the accuracy gain (97.4% against 95.0%).

4.6 Discussion

The edge-based architecture offers clear benefits for fall detection in elderly care. The 72% latency reduction over the cloud deployment is consistent with concerns raised in earlier work (Kumar et al., 2023; Qadir et al., 2023), and every minute of caregiver delay can worsen outcomes for older people living alone. Edge processing is therefore not only convenient but important for time-critical fall response.

Local processing directly addresses the user-acceptance concerns raised by Patel and Ross (2023): because movement data never leaves the home network, the main barrier of off-site data transfer is removed, which could improve adoption among privacy-conscious groups. Offline operation is especially relevant in developing regions such as Nigeria, where homes and communities may not always have reliable internet access, and it supports deployment across varied socioeconomic settings.

The 3.1% false-positive rate, mainly during sudden movements, points to the need for further model improvement or added context such as time of day and user movement patterns. User-specific adaptive thresholding could reduce false alarms and increase recall. The continuous 120 mA draw is a limitation for long-term use without battery management, but duty cycling and deep-sleep modes can extend battery life to days, and nightly charging would be enough for continuous daily monitoring. Finally, the system was tested in a laboratory with both benchmark and primary data, which gives some indication, though not proof, of generalisability to real homes.

5. CONCLUSION AND FUTURE WORK

This paper proposed an edge-based IoT architecture for privacy-preserving fall detection in elderly care. It combines wearable MPU6050 sensors, a NodeMCU microcontroller, a local Flask server running a CNN-LSTM model, and real-time caregiver alerts through a web dashboard and mobile notifications. The main contributions are:

i. A three-layer edge architecture (sensing, edge processing and application layers) designed for privacy-preserving healthcare monitoring.

ii. Quantitative evidence of a 72% latency reduction (4.3 s to 1.2 s) compared with an equivalent cloud deployment.

iii. A privacy-by-design approach that keeps sensitive motion data on the local network.

iv. Experimental validation with 97.4% classification accuracy and 96.9% recall for falls.

v. A demonstration of robust operation during internet outages.

Offline demonstration: Ensuring the system's robustness and reliability in case of internet outages.

The results validate the ability of edge-centric IoT-based approaches to overcome important drawbacks of cloud-based healthcare systems, such as latency due to the network, privacy, and dependence on connectivity to the cloud. This work proposes a hybrid approach using deep learning and in-place processing, making the research towards fall detection systems closer to real world deployable systems**.**

Limitations. The NodeMCU ESP8266 lacks the computing power for on-device deep learning, so the model ran on a local edge server rather than on the wearable. Testing used controlled laboratory conditions and simulated falls, so environmental variability and spontaneous real-world behaviour were not fully captured and performance at home may differ.

Future work. Model compression and quantisation could allow deployment directly on microcontrollers such as the ESP32 using TensorFlow Lite Micro, removing the

external server. Long-term trials in real elderly care settings are needed to assess robustness and acceptability. Physiological and environmental sensors with multimodal learning could provide context and reduce false alarms, and adaptive or federated learning could personalise the model without compromising privacy.

In conclusion, edge-based IoT systems combined with optimised deep learning can deliver low-latency, privacy-preserving and accurate fall detection suitable for real-world healthcare. With the growth of edge AI and TinyML, privacy-preserving local intelligence is likely to shape reliable and ethically responsible monitoring systems.

REFERENCES

Ahmed, N., & Mehmood, R. (2023). Edge-AI frameworks for healthcare IoT: Opportunities and challenges. *IEEE Access, 11*, 42190-42205.

Chen, M., Wang, H., & Yu, L. (2022). A systematic review of wearable sensor-based technologies for fall-risk assessment in older adults. *Sensors, 22*(17), 6752. https://doi.org/10.3390/s22176752

Cong, D. L., Nguyen Quang, B., Truong Minh, D., Truong Cao, D., & Nguyen Ngoc, M. (2024). Continuous wearable-based fall detection using Tiny Machine Learning. *2024 9th International Conference on Applying New Technology in Green Buildings (ATiGB)*, 339-344. https://doi.org/10.1109/ATiGB63471.2024.10717707

Irshad, M., & Alzahrani, A. (2025). Security challenges and analysis tools in Internet of Health Things: A comprehensive review. *Computers, Materials and Continua, 85*(2), 2305-2345.

Karthikeyan, P., Ramesh, S., & Anand, V. (2022). IoT-driven frameworks for remote patient fall monitoring. *IEEE Internet of Things Journal, 9*(22), 22145-22156.

Kumar, S., Gupta, R., & Rawat, D. (2023). Emerging digital technologies used for fall detection in older adults. *International Journal of Medical Informatics, 178*, 104117. https://doi.org/10.1016/j.ijmedinf.2023.104117

Kushwaha, J. P., Kumar, R., Patel, R. B., & Zrar, R. A. (2025). Unveiling IoT ecosystem security: A review of intelligent IDS, trends, challenges, and future directions. *Computers & Electrical Engineering, 122*, 110456.

Mao, W.-L., Wang, C.-C., Chou, P.-H., Liu, K.-C., & Tsao, Y. (2024). MECKD: Deep learning-based fall detection in multilayer mobile edge computing with knowledge distillation. *IEEE Sensors Journal, 24*(24), 42195-42209. https://doi.org/10.1109/JSEN.2024.3456577

Patel, T., & Ross, D. (2023). A review of wearable-sensor-based fall-related recognition systems. *Pattern Recognition Letters, 165*, 21-30.

Prasad, C. R., Moola, R., Satwik, N., Pranith, K., Phaneeshwari, E., & Rao, D. R. (2024). TinyML-based real-time system for detecting falls in elderly people. *2024 Asia Pacific Conference on Innovation in Technology (APCIT)*, 1-6.

Qadir, A., Shen, C., & Wu, Y. (2023). Real-time edge computing for IoT-based fall detection: Review and challenges. *IEEE Internet of Things Journal, 10*(4), 3229-3243.

Salem, B., & Hasan, M. (2024). An effective deep-learning framework for fall detection using acceleration and gyroscope data from wearable sensors. *Journal of Medical Internet Research, 26*, e56750. https://doi.org/10.2196/56750

Shammar, E., Cui, X., Zahary, A., & Alsamhi, S. H. (2025). Threat to trust: A systematic review on Internet of Medical Things security. *Journal of Systems Architecture, 160*, 103456.

Singh, A., & Gupta, S. (2024). Machine-learning approaches for elderly activity recognition: Trends and future directions. *Biomedical Signal Processing and Control, 93*, 105621. https://doi.org/10.1016/j.bspc.2024.105621

Villa, M., & Casilari, E. (2025). Energy-efficient fall-detection system using LoRa and hybrid algorithms. *Biomimetics, 10*(5), 313. https://doi.org/10.3390/biomimetics10050313

World Health Organization. (2023). *Falls: Key facts*. https://www.who.int/news-room/fact-sheets/detail/falls

Yasmin, A., Ngu, A. H. H., Qasem, A., & Queiroz de Farias, M. (2024). Towards improving the real time performance of SmartFall system. *Texas State University Digital Library*.

Zhao, J., Li, Y., & Pan, X. (2022). Quantization-aware training for TinyML-based wearable fall detection. *IEEE Sensors Letters, 6*(12), 450-458.