



**QUEEN'S  
UNIVERSITY  
BELFAST**

## Edge computing transformers for fall detection in older adults

Fernández-Bermejo, J., Martínez-del-Rincon, J., Dorado, J., del Toro, X., Santofimia, M. J., & Lopez, J. C. (2024). Edge computing transformers for fall detection in older adults. *International Journal of Neural Systems*, 34(5), Article 2450026. <https://doi.org/10.1142/S0129065724500266>

**Published in:**  
International Journal of Neural Systems

**Document Version:**  
Publisher's PDF, also known as Version of record

**Queen's University Belfast - Research Portal:**  
[Link to publication record in Queen's University Belfast Research Portal](#)

### **Publisher rights**

© 2024 The Authors.

This is an Open Access article published by World Scientific Publishing Company. It is distributed under the terms of the Creative Commons Attribution-NonCommercial 4.0 (CC BY-NC) License which permits use, distribution and reproduction in any medium, provided that the original work is properly cited and is used for non-commercial purposes.

### **General rights**

Copyright for the publications made accessible via the Queen's University Belfast Research Portal is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

### **Take down policy**

The Research Portal is Queen's institutional repository that provides access to Queen's research output. Every effort has been made to ensure that content in the Research Portal does not infringe any person's rights, or applicable UK laws. If you discover content in the Research Portal that you believe breaches copyright or violates any law, please contact [openaccess@qub.ac.uk](mailto:openaccess@qub.ac.uk).


### **Open Access**

This research has been made openly available by Queen's academics and its Open Research team. We would love to hear how access to this research benefits you. – Share your feedback with us: <http://go.qub.ac.uk/oa-feedback>





## Edge Computing Transformers for Fall Detection in Older Adults

Jesús Fernandez-Bermejo 

*Faculty of Social Science and Information Technology  
University of Castilla-La Mancha  
45600 Talavera de la Reina, Toledo, Spain*

Jesús Martínez-del-Rincon 

*The Centre for Secure Information Technologies (CSIT)  
Institute of Electronics, Communications & Information Technology  
Queen's University of Belfast, Belfast BT3 9DT, UK*

Javier Dorado , Xavier del Toro , María J. Santofimia \* and Juan C. Lopez 

*School of Computer Engineering, University of Castilla-La Mancha  
13071 Ciudad Real, Ciudad Real, Spain  
\*mariajose.santofimia@uclm.es*

Received 19 December 2023

Accepted 19 February 2024

Published Online 16 March 2024

The global trend of increasing life expectancy introduces new challenges with far-reaching implications. Among these, the risk of falls among older adults is particularly significant, affecting individual health and the quality of life, and placing an additional burden on healthcare systems. Existing fall detection systems often have limitations, including delays due to continuous server communication, high false-positive rates, low adoption rates due to wearability and comfort issues, and high costs. In response to these challenges, this work presents a reliable, wearable, and cost-effective fall detection system. The proposed system consists of a fit-for-purpose device, with an embedded algorithm and an Inertial Measurement Unit (IMU), enabling real-time fall detection. The algorithm combines a Threshold-Based Algorithm (TBA) and a neural network with low number of parameters based on a Transformer architecture. This system demonstrates notable performance with 95.29% accuracy, 93.68% specificity, and 96.66% sensitivity, while only using a 0.38% of the trainable parameters used by the other approach.

*Keywords:* Fall detection; older adults; Inertial Measurement Unit; Threshold-Based Algorithm; Transformer Neural Network; Self-Attention; deep learning.

### 1. Introduction

The global increase in life expectancy has brought about new challenges with far-reaching implications. One of these challenges is the risk of falls among

older adults, which significantly impacts health and healthcare systems.<sup>1</sup> The growing interest in fall detection, as discussed in Ref. 2, is driven by the high prevalence of falls in the aging population.

---

\*Corresponding author.

This is an Open Access article published by World Scientific Publishing Company. It is distributed under the terms of the Creative Commons Attribution-NonCommercial 4.0 (CC BY-NC) License which permits use, distribution and reproduction in any medium, provided that the original work is properly cited and is used for non-commercial purposes.

For instance, about 28.5% of older adults, with an average age of 75.4 years, experience falls annually,<sup>3</sup> with subsequent falls being more likely after an initial incident.<sup>4</sup>

Falls among older adults frequently result in unintentional injuries, including fractured hips or wrists and head injuries. As indicated by Robino-vitch *et al.*,<sup>5</sup> falls account for a staggering 90% of hip and wrist fractures and 60% of head injuries in this population group. The financial implications of fall-related injuries are important, with the average cost of a single hospitalization for such injuries among the individuals aged 65 years and older in the US reaching approximately \$17,483 in 2004. Alarming projections suggest that these costs are expected to escalate further, with estimates forecasting a staggering total cost of \$240 billion by the year 2040.<sup>6</sup>

Unattended falls, in particular, can lead to prolonged periods of immobility and distress, increasing the risk of complications. This can result in longer hospital stays and increased healthcare costs.<sup>5</sup> Furthermore, the psychological impact of unattended falls, including fear of future falls and loss of confidence, can lead to reduced physical activity and social engagement, further exacerbating health decline and increasing the risk of future falls.<sup>7</sup>

An automated fall detection system, especially for those living independently, is crucial for timely intervention. While there are existing commercial solutions, such as the Shimmer3 IMU Development Kit<sup>8–10</sup> and the iLife sensor by AlertOne4,<sup>11</sup> their effectiveness is often limited by issues like false positives and reliance on threshold-based detection.<sup>12</sup> The challenge lies in designing wearable devices that balance power consumption, computational resources, and user comfort.<sup>6</sup>

Despite the availability of various solutions, there remains a need for more research to overcome existing limitations.<sup>2,13–15</sup> This work focuses on developing a reliable and user-friendly fall detection device, targeting the safety and well-being of at-risk individuals. Recognizing the distinction between fall detection and fall prediction systems, as highlighted in studies like the one published in Ref. 16, our research emphasizes the critical role of fall detection in promptly managing the aftermath of falls. By addressing challenges, optimizing performance, and refining the design, this research aims to contribute

to the development of innovative, real-time fall detection solutions, which will detect the fall once it has occurred to alert an emergency contact as soon as possible and minimize the negative consequences of a fall. The ultimate goal is to make a significant impact on older individuals' lives by providing enhanced fall detection technologies that promote independence, well-being, and peace of mind, all at an affordable cost.

The main contributions of this paper are as follows: (1) A fall detection system based on an inertial wearable sensor able to detect and communicate falls in real time with minimal interference and high versatility. (2) A novel portable device that comprises both sensors and customizable AI processing on the edge to detect falls in real time, while preserving extended battery life. (3) A novel lightweight Transformer architecture that provides state-of-the-art performance on fall detection, while fitting in the proposed device architecture due to its small parameter size. This architecture achieves a 99.6% reduction on parameter size while preserving the detection accuracy. (4) A comprehensive system evaluation using three datasets to validate the performance of our proposed approach.

The paper is structured as follows. First, the state-of-the-art for fall detection systems is presented in Sec. 2 from the perspectives of both the hardware and software solutions. Then, Sec. 3 describes the proposed solution and Sec. 4 presents the different experiments carried out to validate the proposed solution. Finally, Sec. 5 presents the main conclusions drawn from the conducted study.

## 2. Related Works

### 2.1. Hardware approaches for fall detection

The state-of-the-art in fall detection and prevention systems can be broadly categorized into three types of hardware solutions: environmental-based sensors, vision-based systems, and wearable-based sensors.<sup>17</sup>

Environmental-based sensors are typically stationary, located in specific areas of a user's home. These sensors can vary widely, from pressure and vibration sensors to presence sensors. However, they are easily affected by external environmental factors, which can lead to false alarms. For instance, a non-wearable ultra-wide-band (UWB) sensor installed in

the ceiling to monitor the activities underneath its area of action was proposed in Ref. 18. Similarly, the work in Ref. 19 explored the use of Doppler radar for human fall detection. The study suggested that while the number of false alarms may still be high for practical use, integrating the radar fall detection system with other sensor modalities such as acoustics and cameras could considerably reduce false alarms. The work in Ref. 20 proposes a particle filter and combines the depth and thermal information based on the velocity and position of the head to detect falls. In general, despite the potential of such systems, their main limitation is their susceptibility to environmental influences, which lead to a high level of false alarms. Furthermore, the detection is only limited to the area covered by the environmental sensor.

Systems based on visual recognition utilize diverse techniques in computer vision for identifying falls or analyzing walking patterns to assess an individual's fall risk. Such systems can incorporate various camera types, from a solitary RGB camera to setups involving multiple cameras,<sup>21–24</sup> and depth cameras,<sup>25</sup> such as the Kinect.<sup>26,27</sup> Nonetheless, the primary challenge encountered by these approaches is the compromise of user privacy, coupled with the limitation that these systems are only effective in areas within the camera's field of view. Nevertheless, there is a new line of research that puts the focus on privacy preservation based on the use of thermal sensor arrays. In this sense, the work in Ref. 28 proposes a novel human-in-the-loop fall detection system employing a low-resolution thermal sensor array, addressing privacy concerns in personal environments. Their approach, based on motion sequence classification with a Recurrent Neural Network (RNN), demonstrated a remarkable accuracy of 99.7% in detecting human falls, showcasing the potential of thermal sensors in maintaining privacy while ensuring high detection accuracy. Furthermore, the eHomeSeniors dataset, presented in Ref. 29, offers a unique resource for fall detection research using privacy-friendly infrared thermal sensors. This dataset is notable for including data from both young volunteers and performing artists trained to emulate the fall conditions of older adults, providing a more realistic and varied set of data for algorithm development.

Wearable sensors are based on devices placed on the user's body which means that they are not limited to the area covered by a static monitoring

device. These are commonly the Inertial Measurement Units (IMUs), consisting of accelerometers and gyroscopes that measure the user's movements. In this category of sensors, smartphones have emerged as particularly significant due to their robust processing capabilities, coupled with the integration of accelerometers and gyroscopes. Additionally, there are solutions that utilize bracelets or bands tailored for the wrist, waist, or ankle, as well as those incorporating sensors into smart clothing, alongside smartphone-based systems.<sup>30–35</sup> However, a significant limitation of wearable sensor-based solutions is that they often require the user to wear the device in a specific location, which can be inconvenient or uncomfortable. The adoption of a wristband for fall detection is highlighted for its discrete nature, as noted in Ref. 36, while the pursuit of minimal intrusiveness inspired the creation of a device integrated into footwear, as described in Ref. 37. The research in Ref. 38 introduces a smart clothing-based approach, utilizing gravitational acceleration data for fall detection via a hidden Markov model, as explored in Ref. 39. Differing from this, the study in Ref. 40 offers greater flexibility, eliminating the need for sensor placement in a fixed position. Instead, the suggested IMU sensor is adaptable for placement on any body part. Although the obtained accuracy for fall detection is promising, the system is not yet ready to work in real time. The primary drawback of mobile phone-based methods, as noted in Refs. 41 and 42, is their reduced practicality indoors. While carrying a phone outdoors is convenient, it is less common for individuals to keep their phones in their pockets when inside. In this sense, the work in Ref. 42 demonstrates the feasibility of a real-time, deep-learning-based fall detection system directly on a smartwatch, using a collaborative edge-cloud framework. This approach addresses the practical limitations of smartphone dependency by leveraging the smartwatch's capabilities for both data sensing and processing, thereby ensuring continuous monitoring and immediate response in the event of a fall, even when the user is not in close proximity to their phone.

Building upon these existing approaches, the work proposed here introduces a novel dimension in wearable-based fall detection. This sensor is equipped with a custom-designed, energy-efficient neural

network architecture that operates in real time, directly on the wearable device. This architecture not only ensures immediate fall detection and alerting but also significantly reduces the dependency on continuous smartphone connectivity, addressing a key limitation highlighted in previous studies such as Ref. 42. Furthermore, our system uniquely integrates advanced machine learning (ML) algorithms, including a Transformer-based model, which significantly enhances detection accuracy while maintaining a small footprint suitable for edge computing. This approach represents a significant advancement in the field, combining the benefits of high accuracy, real-time processing, user convenience, and reduced power consumption, thereby making it a highly effective solution for continuous fall monitoring in various settings.

## 2.2. Algorithmic and architectural approaches in fall detection

The field of fall detection systems has witnessed a growing interest in leveraging neural networks and deep learning techniques. However, the revision of the state-of-the-art brings into light that some of these studies lack sufficient information to confirm real-time detection capabilities, particularly when integrating in embedded and edge devices. Furthermore, a majority of these works have yet to undergo extensive testing in real-world or *in-the-wild* scenarios.

The main challenge faced when embedding a fall detection algorithm into a wearable device lies in adapting typically large models to devices with limited resources. In this context, the advancements in Transformer Neural Networks and deep learning have enabled a step change. In this sense, the deep Transformer model presented in Ref. 43 and the Graph Transformer Network in Ref. 44 demonstrate the potential of these architectures in handling complex, sequential data efficiently, which is crucial for real-time fall detection. Similarly, the work in Ref. 45 for patient-independent seizure detection employs Convolutional Neural Network (CNN)–Transformer models and provides insights into handling diverse datasets effectively. Moreover, the deep learning approaches in medical imaging and diagnosis, as explored in Refs. 46 and 47, show how pretrained Deep Convolutional Neural Networks can be optimized for specific tasks, offering a blueprint

for adapting such models to wearable devices. These studies collectively underscore the evolving landscape of neural network applications, where the challenge extends beyond model accuracy to fitting advanced algorithms into compact, wearable formats. This body of work provides a foundation for our approach to implementing a deep-learning-based fall detection system that is both efficient and effective, even within the constraints of a wearable device.

In this sense, the work in Ref. 41 proposes a deep learning approach based on the information provided by thermal sensors. The architecture is comprised of convolutional, max pooling, and Recurrent Neural Network layers, specifically Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), or Bi-LSTM. The system achieved the accuracy, sensitivity, and specificity of 93% using Bi-LSTM. Nonetheless, the tool was not validated in real time in home setups. The work in Ref. 48 proposes a vision-based fall detection system based on curvelet transforms for feature extraction and Support Vector Machine and HMM for posture classification and activity recognition. The system achieved an accuracy of 96.88% and an  $F$ -measure of 0.96 using the HMM–Support Vector Machine algorithm. In Ref. 49, a specialized Convolutional Neural Network (FD-CNN) was used for fall detection. The system was designed for low-power sensing, making it suitable for real-time applications and devices with limited computational resources, indicating the potential for edge computing. The work in Ref. 50 proposes a real-time patient monitoring framework for fall detection that uses an LSTM model for machine learning and an edge computing framework for real-time fall detection. Moreover, a series of studies are also observable where Recurrent Neural Networks are employed for fall detection, treating inertial measurements as a sequence, much like the approach undertaken in this work.<sup>51–53</sup>

Despite these advancements, there are several limitations to consider. For instance, the system in Ref. 41 has a number of false positives due to the increase of temperature in objects, indicating potential issues with objects having temperatures higher than room temperature. The system performance of Ref. 48 could be affected by shadows or partial occlusion of body parts. In Ref. 49, user

acceptance tests were not conducted, and the use of the proposed technology was limited to an adapted vest. In Refs. 51–53, Recurrent Neural Networks do not provide significant advantages over other available architectures. They typically occupy more memory space, rendering them impractical for execution at the edge.

The work proposed here presents several novel elements that significantly advance the field of fall detection. First, the proposed neural network architecture is specifically tailored for efficient operation on wearable devices with limited computational resources. This architecture, unlike the general-purpose networks used in previous studies, is optimized for the specific task of fall detection, ensuring high accuracy with minimal memory and power requirements. This makes it particularly suitable for real-time applications and continuous monitoring *in-the-wild*, addressing a critical gap in existing research. Second, the proposed approach goes beyond traditional deep learning techniques by incorporating a Transformer-based model that not only is more compact than typical Recurrent Neural Networks but also offers higher performance in terms of accuracy and speed, making it ideal for deployment in edge computing scenarios. This is a significant improvement over the LSTM and GRU models used in studies like Refs. 41 and 50, which, while effective, are often too resource-intensive for wearable devices.

### 3. The Proposed System for Real-Time Fall Detection

This work proposes a fall detection system with a strong emphasis on versatility. A prevalent issue in many existing systems, as detailed in the literature, is their limited adaptability in various settings, particularly when transitioning between indoor and outdoor environments. For example, systems based on environmental sensors are restricted to the specific area of sensor deployment, making them ineffective when the user moves outside this zone. This constraint is also inherent in camera-based systems, which additionally impacts user privacy therefore resulting in low user acceptance.

Wearable sensors, while promising, are not without their challenges. Some systems rely on Fog

processing via a gateway, which inherently restricts the operational range based on the maximum distance for data transmission between the sensor and the gateway. Typically, these systems use a static BLE (Bluetooth Low Energy) gateway, specifically Class 2 Bluetooth, which provides a communication range of approximately 20 m. This confines the fall detection service to a static 20-m radius, making the system nonfunctional if the user moves beyond this distance, or if the system is utilized outdoors. Alternatively, some wearables utilize smartphones with Internet access to transmit IMU data for cloud processing, thereby overcoming the static limitation. However, these systems need that the user carries the phone in a specific location, which if not adhered to, can lead to incorrect system inferences. While these systems perform well outdoors, their reliability diminishes in indoor settings, such as homes or nursing homes, as they require users to constantly carry the phone, which is an unrealistic expectation.

To mitigate these challenges, we propose a system that integrates a wearable device with a smartphone or tablet. The wearable sensor is designed to capture inertial data from the user's body, specifically focusing on acceleration and angular velocity. The sensor's firmware incorporates a neural network that processes this data to determine whether the recorded measurements correspond to a fall or an Activity of Daily Living (ADL). Upon detecting a fall, the sensor communicates with the mobile device (either a smartphone or tablet) via BLE, prompting the device to automatically place a call to a designated emergency contact to report the detected fall. This proposed system, despite utilizing BLE, ensures versatility in both outdoor and indoor environments having a movable gateway. Outdoors, the user can freely carry the sensor and smartphone without any specific placement requirements for the phone. Indoors, the user is not required to constantly carry the smartphone, as the Bluetooth coverage is typically sufficient to encompass almost the entire house.

The following subsections will provide a detailed description of the system, covering the hardware platform used, the neural network architecture, the software architecture for inference and communication, and the user application.

### 3.1. The hardware for real-time fall detection

In the field of fall detection, the MetaMotionR<sup>a</sup> and Puck.js<sup>b</sup> sensors have been highlighted in previous studies as promising devices due to their unique capabilities.<sup>54</sup> However, upon closer examination, we identified certain constraints. For instance, the MetaMotionR sensor is designed for developer-friendly use and is equipped with a feature-rich firmware. This firmware includes a variety of functionalities such as logging capabilities, the ability to stream data to other devices, substantial data storage, and a software stack designed to work in tandem with the sensor. However, a significant limitation of this sensor is the inability to modify or reprogram its firmware, which poses a considerable challenge for developers aiming to create flexible applications that operate in real time. The application must continuously stream data to a gateway for user monitoring, which is impractical due to range limitations and potential connection losses.

On the other hand, the Puck.js sensor aims to offer more flexibility by allowing partial firmware reprogramming. Specifically, this device's firmware includes a Javascript interpreter, enabling modifications to certain device behaviors. For instance, it allows customization of the information the sensor sends via Bluetooth and provides the option to select the data to send, eliminating the need for continuous streaming as with the MetaMotionR sensor. While this offers a degree of freedom, it is still restrictive, as many aspects remain unmodifiable. Additionally, the integrated interpreter inherently consumes a significant portion of the device's memory.

To address these challenges and achieve greater flexibility, this study opted to develop a fit-for-purpose wearable device with fully modifiable firmware. The wearable device was built using the Nicla Sense ME<sup>c</sup> device, part of Arduino's new PRO range aimed at bridging their products to the industry. Notably, the sensor features an nRF52832 chip, providing Bluetooth connectivity and housing the primary ARM Cortex M4 microcontroller. To monitor user movements, the device incorporates a state-of-the-art IMU sensor by Bosch, the BHI260AP. This sensor integrates an accelerometer with a range

of  $\pm 16G$  and a gyroscope with a range of  $\pm 2000^\circ/s$ , the reason for using these ranges is their prevalence in the majority of relevant datasets, such as the SisFall<sup>55</sup> dataset. It also includes a 32-bit Fuse2 microcontroller with software functionalities to facilitate various Artificial Intelligence tasks. The two main components of the Nicla sensor, the nRF52832 chip and the BHI260AP IMU, communicate with each other through an SPI (Serial Peripheral Interface) interface, which facilitates access to the services offered by the BHI260AP chip. The Nicla Sense ME device provides substantial flexibility due to the fully programmable firmware of the nRF52832 chip. This allows for direct reading of IMU data and the selection of the appropriate data processing method. The primary application of this feature is to integrate all processing within the wearable device's firmware, enabling edge inference and minimizing data transmission to only instances when a fall is detected.

In order to enhance the user-friendliness of the device, several modifications were implemented. Initially, a battery was connected to provide power to the wearable device when it is worn by the user. A 3.7-V LiPo (Lithium Polymer) battery with a capacity of 400 mAh was selected for this purpose, and it was directly soldered to the appropriate power pins on the board. Following this, a compact enclosure was designed to comfortably accommodate both the battery and the wearable device when positioned on the user's waist. The final design of the wearable device is illustrated in Fig. 1, while Fig. 2 provides a depiction of its internal structure. The dimensions of the sensor are a width of 31.55 mm, a height of 42.75 mm, and a depth of 17.7 mm.

There is much debate about the optimal location for fall detection. Devices on the wrist are often more comfortable for the user but tend to yield poorer results. Waist-worn devices, despite potentially being somewhat less comfortable, can be sufficiently ergonomic and provide better results in fall classification if they are small enough. In a study,<sup>56</sup> it is confirmed that, among all possible locations, the wrist yields the worst results, while the waist is the most effective. This position may pose a challenge for some users, for instance, if the user is wearing a

<sup>a</sup><https://mbientlab.com/store/metamotionr/>.

<sup>b</sup><https://www.puck-js.com/>.

<sup>c</sup><https://store.arduino.cc/products/nicla-sense-me>.



Fig. 1. Fall detection wearable device.

specific type of garment that would require an external element to secure the sensor to the body. Although the system can be used in outdoor environments, it is designed for indoor use where no extra elements would be required. However, the use of straps to attach the sensor is an option for such situations.

### 3.2. The software system for fall detection

To enable fall detection, the wearable device needs to be equipped with the capability to identify falls accurately. To achieve this, the optimal location to place the IMU device has to be determined. The literature offers various options for fall detection placement, including the foot, calf, thigh, wrist, hip, and chest. After careful consideration, the hip was selected as the most suitable position for effectively differentiating falls based on inertial measurements, as demonstrated in a previous study.<sup>57</sup>

The classification algorithm comprises two primary components: a Threshold-Based Algorithm (TBA) responsible for screening potential fall movements, and a neural network-based model. The neural network receives the movements filtered by

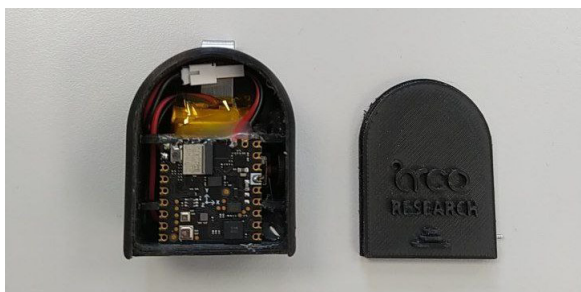


Fig. 2. Inside view of the fall detection wearable device.

the TBA and performs inference to classify each movement as either a fall or an Activity of Daily Living.

We emphasize that both algorithms presented here, i.e. both the TBA and the neural network, operate entirely and exclusively on the device that users will carry. Thus, this sensor has the capability to collect inertial measurements from the user, perform preprocessing, use the TBA to detect potential falls, extract a time window representing a movement, and once a fall is detected, process this window in the neural network and communicate the occurrence of a fall through a notification via BLE.

#### 3.2.1. Threshold-based algorithm

TBA is one of the most popular algorithms for fall detection in devices with computational constraints. Being much simpler than the ML-based algorithms, as they only compare the sensor's measurements with a reference value, TBAs are ideal for such devices. They are less computationally demanding and consume less energy, making them suitable for edge processing. However, this advantage comes with a trade-off, as TBA algorithms are less precise and more prone to false positives compared to the ML-based approaches.

However, due to the higher energy consumption of ML algorithms for continuous monitoring, TBAs are well suited to serve as a preliminary filter, conserving computation by discarding movements not deemed potential falls. In this study, the TBA is used to identify movements susceptible to being falls. Subsequently, the identified movements are sent to the neural network for classification into either an ADL or a fall.

Given the use of inertial measurements, our TBA utilizes the Signal Vector Magnitude (SVM) of acceleration, a feature previously employed in the literature.<sup>58,59</sup> The SVM of acceleration is computed as follows:

$$\text{SVM} = \sqrt{\text{Acc}_X^2 + \text{Acc}_Y^2 + \text{Acc}_Z^2}, \quad (1)$$

where  $\text{Acc}_{\text{axis}}$  is the acceleration in each of the three coordinates axis  $\in [X, Y, Z]$ .

The reference threshold used for comparison is set at approximately  $\text{Th} = 3G$ . This value is considered appropriate based on previous studies<sup>60</sup> as it



effectively filters movements resembling potential falls. A lower threshold would result in sending a considerable amount of data to the neural network, consuming excessive energy and resources, while a higher threshold increases the chances of false negatives, potentially discarding mild falls.

Once the threshold is exceeded, i.e.  $SVM \geq Th$ , indicating a potential fall, the system identifies and forwards the detected movement to the neural network for further classification. To ensure a reliable process, it is necessary to have a precise definition of what qualifies as a “movement” in this context. In the subsequent part of our algorithm involving the neural network, a “movement” is defined as a sequence of inertial measurements. These measurements encompass both acceleration and angular velocity data in the three axis coordinates, captured by both the accelerometer and gyroscope, respectively. Each sequence is confined within a 1-s time window.

The choice of the 1-s-duration windows is based on the average fall duration of approximately 1.77 s.<sup>61</sup> By using a slightly shorter window, we exclude the “flight phase” of the fall. The “flight phase” occurs when the user is still airborne before hitting the ground. The emphasis on this phase is secondary, as the system’s primary goal is not fall prevention but rather the detection and assistance of falls after they have occurred. This particular phase is not accounted for in the analysis window, as the triggering threshold is expected to activate when the subject makes contact with the ground. On the contrary, taking into account that the “flight phase” constitutes half of the fall time, the remaining fall

duration is about 0.88 s, which the 1-s window effectively captures. Moreover, limiting the window’s size optimizes memory efficiency, a critical aspect when deploying machine learning algorithms on edge devices. Minimizing resource consumption allows for better overall performance. Expanding the window would increase memory usage and the neural network’s input dimensions, which we aim to avoid. Instead, we prioritize obtaining the most relevant information with minimal data.

To achieve this goal, we will develop a state machine inspired by a previous study.<sup>62</sup> This state machine will encompass different states, including Sampling Data, Post-Peak, Post-Fall, and Check-Fall, to effectively capture the pertinent data within the defined 1-s window.

This state machine can be observed in Fig. 3, where each state represents the following: (1) *Sampling*: The system is capturing data at a frequency of 50 Hz. (2) *Post-Peak*: A significant acceleration, greater than 3G, is detected. Subsequently, acceleration and angular velocity data are saved in a window. (3) *Post-Fall*: Following the detection of a peak acceleration, all the data representing a movement within a 1-s window is captured. The state machine then enters a waiting period of 1.5 s. During this time, if another acceleration greater than 3G is detected, the system cleans the window and transitions back to the *Post-Peak* state. Otherwise, the system proceeds to the *Check-Fall* state. (4) *Check-Fall*: In this state, the window’s data is sent to the machine learning model for inference. If the inference yields false, indicating the detection of an ADL, the system returns to the *Sampling* state. Conversely, if

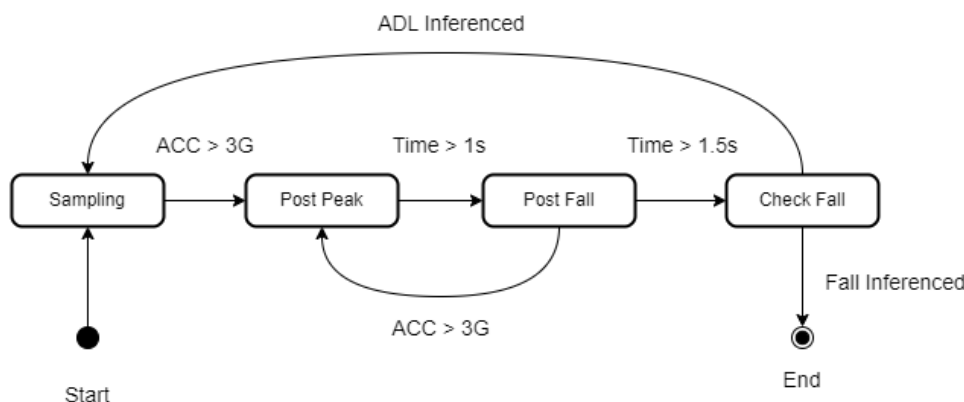


Fig. 3. State machine for the TBA.

the inference yields true, indicating a fall event, the system notifies a third party, i.e. a mobile phone, tablet, etc., of the occurrence.

This state machine is a modification of the one present in Abbate, removing the Activity Test state to reduce the computational resource usage.

An important aspect to address when capturing the window is the data recording frequency. Again, due to the device's limitations, an appropriate frequency must be chosen to avoid excessive data and prevent information loss. Following the work of Ref. 63, a frequency of 50 Hz has been selected since human body movements possess components with frequencies up to 15 Hz. Thus, capturing data with at least 30 samples per second should be enough.

In order to analyze the accelerations captured by the window, a 3D representation was conducted where the captured points from two distinct movements, a free fall and an ADL, can be observed along each of the X-, Y-, and Z-axes (Fig. 4). Contrary to the intuition, significant accelerations do not necessarily always occur along the Z-axis, thus justifying the use of SVM for this magnitude. Additionally, a clear separation is observed in Fig. 4 between accelerations associated with an ADL (dots in orange) and a fall (triangles in blue).

Considering the explanations above, the TBA filters movements susceptible to being falls, capturing a window of 50 measurements, comprising acceleration measurements in all three axes and angular velocity measurements in all three axes. Therefore, each measurement consists of six

dimensions or channels, resulting in a total of 300 data points for each movement. This sequence comprising 300 data points represents the movement and will be inputted to the neural network for inference to determine if a fall has been detected or not.

### 3.2.2. Lightweight transformer neural network

After the TBA identifies a movement as potentially indicating a fall, the next step in the fall detection algorithm involves verifying whether it is indeed a fall or an ADL. To accomplish this, machine learning techniques are employed. In the literature, various approaches have been explored, ranging from traditional algorithms like Support Vector Machines and Decision Trees to neural network-based solutions. However, the trend has shifted toward using neural networks more frequently, primarily due to their ability to autonomously learn essential data features during training. In contrast, traditional models require developers to manually select features, which can introduce errors due to human intervention.

There is a wide variety of neural network architectures used in diverse fields, including Convolutional Neural Networks extensively employed for image processing, Recurrent Neural Networks designed for sequence or time-series data, and the more recent Transformer models, as seen in GPT-4. For this study, the fall is considered as a time sequence, where each measurement represents the body state of a person at a specific time instant. Therefore, Recurrent Neural Networks, specifically the LSTM or GRU models, were considered due to their ability to mitigate information loss over sequential data, which is often observed in the traditional RNNs. However, the challenge lies in developing a lightweight neural network that can be executed on the edge, considering the memory and computational constraints it entails. TensorFlow Lite, a widely used framework for deploying neural networks on microcontrollers, was selected for development. Nevertheless, this framework has limitations in terms of available layers for edge devices, and as of now, Recurrent Neural Network layers are not fully implemented in TensorFlow Lite, primarily due to their high memory cost.

Given this limitation, an alternative approach was pursued, developing a novel lightweight architecture inspired by the Transformer architecture,<sup>64</sup>

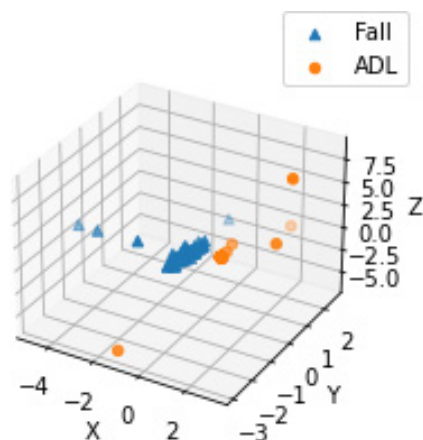


Fig. 4. Representation of the accelerations of a free fall and an ADL in a 3D space.

with a focus on Self-Attention. Since there is no need to reconstruct the original input from the data, only the encoder part of the Transformer is required. This part needs to be reduced in size, as an excessively large neural network would not be feasible for the Nicla Sense ME, where it will be deployed.

Given the challenge of developing a lightweight neural network for edge execution, a critical aspect of our design process was the selection of hyperparameters, particularly the number of neurons in each layer. This selection was achieved through an iterative process of trial and error, focusing on minimizing the network size while maintaining high accuracy. Our approach was guided by the constraints of the edge computing environment and the need for efficient processing.

To systematically explore the hyperparameter space, we employed tools like Keras Tuner. This allowed us to define ranges for hyperparameters based on the limitations of our computational resources and the requirements of the fall detection task. Through this process, we conducted a search within these ranges, and the resulting combination of hyperparameters was the one that yielded the best results in terms of accuracy and model efficiency. This method ensured that our model was not only effective but also optimized for the specific constraints of the Nicla Sense ME device.

The input to the neural network is a set of 50 measurements, with each measurement consisting of six dimensions representing the  $x$ -,  $y$ -, and  $z$ -components of acceleration and angular velocity. This results in an input vector with a dimension of  $50 \times 6$  elements. Before these measurements are fed into the neural network, a preprocessing step is conducted. To ensure that no specific element dominates the model, each data feature is normalized using a standard scaler. This normalization is crucial because neural networks are sensitive to the magnitude of input values during training. Without this preprocessing step, the larger values of angular velocities could disproportionately impact the results, potentially overshadowing the significance of other features.

After normalizing the data, a technique called positional encoding is applied to represent the position of each data point in the sequence. This positional encoding method is inspired by the Transformer architecture introduced in the paper by

Vaswani *et al.*<sup>64</sup> In this approach, we treat each of the 50 samples in the sequence as if it were a “word” in the context of natural language processing. In natural language processing, a “word” is a fundamental unit of language that conveys meaning and context. It is typically represented as a dense vector, known as “word embedding”, in a continuous vector space. Word embeddings capture semantic relationships between words, allowing algorithms to understand the context and meaning of individual words based on their vector representations. In our context, each sample in the sequence of 50 elements, containing six inertial data elements, is treated as a “word”. The six inertial data elements act as the “input embeddings” of the “word”.

By using positional encoding and this “word” and “input embeddings” analogy, our neural network gains the ability to understand the temporal relationships and dependencies between the measurements in the sequence. This comprehensive understanding allows the network to make accurate predictions based on both acceleration and angular velocity components, optimizing its ability to distinguish falls from other activities. In essence, this method enables the neural network to consider the sequential nature of the data and capture the important patterns that contribute to fall detection.

The input to the neural network is a  $50 \times 6$  vector, where each element corresponds to a specific data point (a “word”) in the sequence. To facilitate processing, this  $50 \times 6$  vector is then flattened into a single 300-element input tensor.

The neural network’s architecture is designed once the input data is established. To address memory constraints of the device, the initial step involves significantly reducing the input vector’s dimensionality. This compression is achieved through an encoder layer utilizing a sigmoid activation function, inspired by Ref. 65. The encoder layer condenses the input from 300 to 10 float elements. While this is a substantial reduction for a single layer, adding more encoder layers would be impractical given the device’s limitations.

Next in the architecture is the attention layer. It is responsible for calculating the key, query, and value vectors, which are used to generate the attention vector. This vector is then further processed through a linear dense layer, reducing its dimension

to just three elements. The data is then passed through a sigmoid layer, which produces an output value between 0 and 1. If the output value is less than 0.5, it indicates that the data represents an ADL, while a value greater than 0.5 signifies a fall event. A graphical depiction of the neural network can be found in Fig. 5.

### 3.3. Network training

Once the neural network is constructed, it is essential to train it with data to accurately classify falls and ADLs. For this purpose, it will be necessary to manipulate the data through a series of phases.

The data first undergoes a preprocessing stage to ensure optimal neural network training. Initially, irrelevant data from the dataset, such as timestamps, are discarded. Subsequently, if the data has not been captured at a frequency of 50 Hz, a resampling must be performed for each data point. After resampling, each movement is represented by a  $50 \times$

6 data array. It is crucial to scale the data to ensure no single feature disproportionately influences the training. This scaling is particularly important since angular velocities are much larger measurements than accelerations and would thus have a more significant impact on the outcome. As previously mentioned, Self-Attention is employed, necessitating the addition of position encoding to the data. Drawing inspiration from the study by Vaswani *et al.*,<sup>64</sup> specific inertial measurements are treated as words, with the  $x$ -,  $y$ -, and  $z$ -components of acceleration and angular velocity acting as word embeddings. The formulas used to add position encoding to each of the 300 data points in a sample are given by Eqs. (2) and (3),

$$PE_{(\text{pos},2i)} = \sin(\text{pos}/10000^{2i/d}), \quad (2)$$

$$PE_{(\text{pos},2i+1)} = \cos(\text{pos}/10000^{2i/d}). \quad (3)$$

In these formulas,  $\text{pos}$  denotes the components of acceleration and angular velocity captured at a specific instant in time. The variable  $i$  represents the order of each of the six measurements, while  $d$  represents the total number of sample components, in our case six, the  $x$ -,  $y$ -, and  $z$ -components of acceleration and angular velocity. Given that our sample consists of the  $x$ -,  $y$ -, and  $z$ -components of acceleration and angular velocity, this number  $d$  will be 6. Thus, within a specific inertial measurement, if the component occupies an even position, Eq. (2) is used, and if it occupies an odd position, Eq. (3) is applied.

The selection of an appropriate threshold in this six-dimensional space is a critical aspect of the proposed methodology. It is not a straightforward process of choosing a single value but rather involves a multi-dimensional analysis. The threshold determination is based on a combination of empirical analysis and optimization techniques. An analysis of the dataset is empirically conducted to understand the typical ranges and patterns of the six measurements during various activities, including falls and nonfall movements. This analysis enables the identification of preliminary threshold values that can distinguish between fall and nonfall events. Following the empirical analysis, optimization techniques are employed to refine these thresholds. This involves iteratively testing different threshold values and assessing their impact on

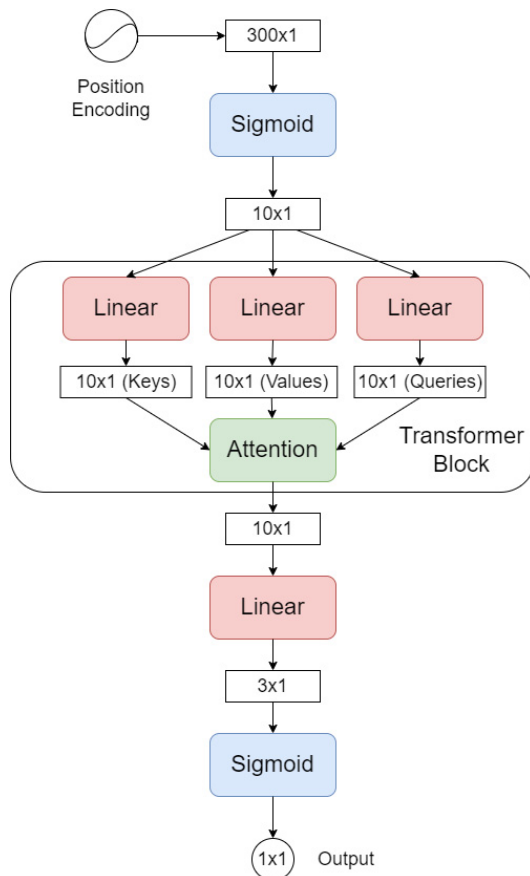


Fig. 5. Neural network proposed.

the system's ability to accurately detect falls. The optimization criteria include maximizing the detection accuracy (true positives) while minimizing false alarms (false positives). The selected thresholds are then integrated into our fall detection algorithm. This algorithm uses the thresholds to analyze the incoming six-dimensional data in real time, determining whether a fall has occurred.

Finally, position encoding is integrated into our fall detection algorithm alongside the optimized thresholds. This combination allows us to effectively analyze the six-dimensional data in real time while considering the temporal context provided by position encoding. The proposed approach employs position encoding to enhance the effectiveness of the neural network, particularly in the context of analyzing time-series data from accelerometers and gyroscopes. The primary purpose of incorporating position encoding is to provide the neural network with contextual information about the sequence and timing of the data points. This leads to improved accuracy and reliability in the model's predictions.

To conclude the preprocessing, the input vector is flattened to be introduced into a neural network with an input tensor of 300 elements, transforming our array from a  $50 \times 6$  shape to a  $300 \times 1$  shape.

Next, we move to the training process, where the model is fitted with the data selected for training. To assess the model, various training sessions have been conducted using diverse datasets. In the experiments section, one can observe the different models that have been trained.

The final phase of training involves model quantization, a critical step in addressing the memory constraints associated with edge processing. Given that the proposed system is designed to operate on resource-constrained devices, such as the Nicla Sense ME sensor, which must also manage libraries for tasks like reading the inertial measurements and BLE communication, optimizing the model's size turns into an essential requirement. To start, the TensorFlow Lite model converter will be employed to create a smaller model that is well suited for execution on microcontrollers. The conversion process retains the model's essential functionality while significantly reducing its size. Following the model conversion, the quantization tasks proceed. This involves modifying the network to use 8-bit integers for

representing inputs, weights, and outputs. By doing so, the memory footprint of the neural network is dramatically reduced. This step aligns with industry best practices for optimizing models for edge devices. Once the model is quantized, a hexadecimal dump of the model is generated. This dump contains the quantized model's information and structure. This format is suitable for loading the model onto the Nicla Sense ME sensor, ensuring that it can efficiently carry out fall detection tasks while operating within its memory constraints.

To minimize the size of the neural network, once the model is trained, a conversion process will be carried out. During this process, a lightweight model will be generated from the initially created model. In addition to this conversion, all elements of the network, including input and output elements, as well as neuron weights, will undergo a quantization process. The goal is to ensure that each of these elements occupies only 8 bits. First, this process involves using the TensorFlow Lite model converter to obtain a smaller model suitable for execution on a microcontroller. Following that, the quantization tasks are performed, modifying the network so that inputs, weights, and outputs are represented using 8-bit integers, thus reducing our network's size. Once the model is quantified, a hexadecimal dump of the model is created to load it onto the device.

### 3.4. System architecture

An essential feature of this system, as noted earlier, is its versatility across a wide range of environments, encompassing domestic settings, nursing home facilities, and even outdoor areas. To cater to these diverse scenarios, we propose an adaptable architecture that can flexibly adjust to different scenarios.

The main component of the system is the wearable device with the proposed neural network implemented within it. This component alone is capable of detecting falls, but once a fall is detected, it is necessary to notify a third party who can assist the user. To this end, an Android application has been developed that will make a call using the Telegram network when a fall is detected by the wearable device. The decision to develop an Android application was based on the fact that most portable systems today (mobiles and tablets) use Android as their operating system.

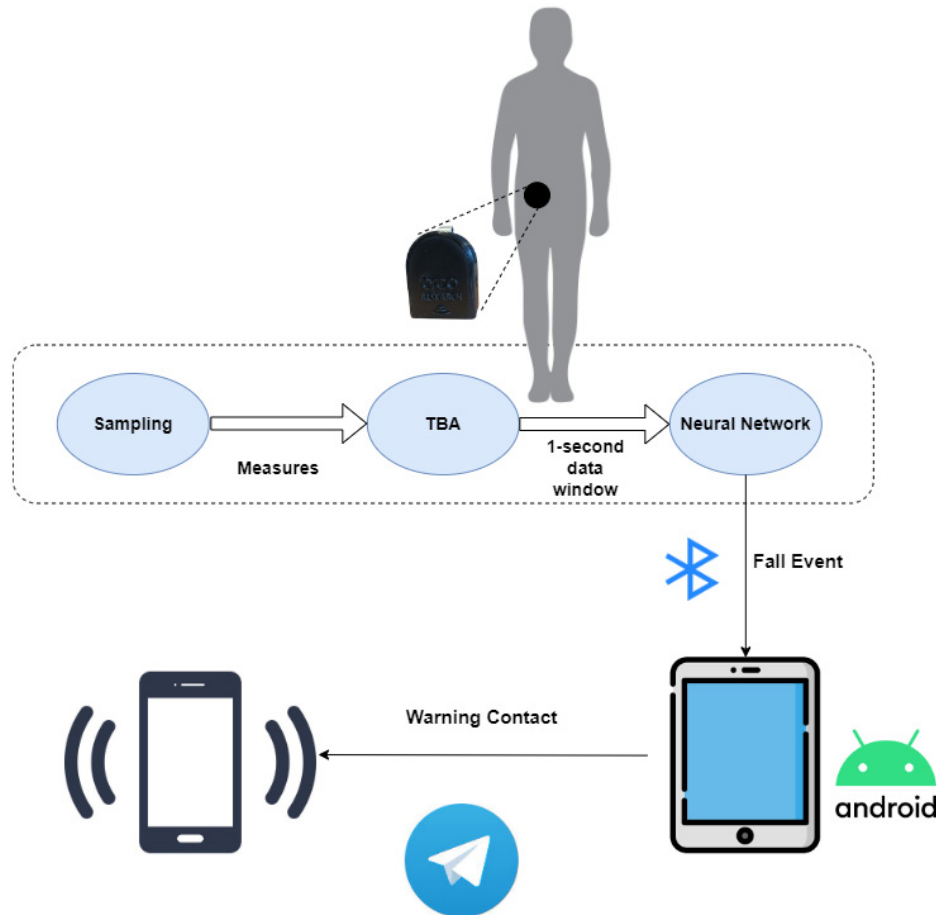


Fig. 6. System diagram.

Under this system, the Nicla Sense ME sensor is tasked with continuously monitoring the user's inertial measurements. Upon TBA detecting a movement that could potentially indicate a fall, it captures a 1-s data window and forwards this to the neural network. The neural network processes this data and determines whether the movement signifies a fall. In the event of a fall, the wearable device uses BLE to broadcast a fall detection alert, effectively acting as a beacon. This alert is picked up by an Android device with the installed application. Upon recognizing the alert, the application initiates a call via Telegram to the designated emergency contact. A diagram of the system is presented in Fig. 6. One can watch a video demonstration at the link provided.<sup>d</sup>

<sup>d</sup><https://www.youtube.com/watch?v=cwGLdhaAN6U>.

## 4. System Validation

### 4.1. System assessment

The effectiveness of the lightweight model proposed here will be assessed through various experiments in which the effectivity of the neural network will be evaluated, including comparisons with other models from the state-of-the-art. To conduct both the evaluation and the comparisons, three different metrics are proposed: accuracy, specificity, and sensitivity. Accuracy will calculate the percentage of correct classifications over all movements that have triggered the TBA. Sensitivity will indicate the proportion of falls correctly identified by the model as actual falls. Conversely, specificity will denote the percentage of ADLs correctly recognized as such by the system. All the data used to evaluate the model is

Table 1. Comparison of models trained on Ref. 62 and tested on Ref. 54 datasets.

Work	Accuracy	Specificity	Sensitivity	Location	Algorithm
Our full approach	<b>95.75%</b>	<b>99.38%</b>	<b>93.07%</b>	Waist	Self-Attention
Ref. 54	93.51%	95.45%	92.04%	Waist	Support Vector Machine

the one that is considered as a possible fall for the TBA.

The first experiment consists in comparing our model with the study in Ref. 54, wherein the proposed model will be trained and tested using the identical datasets employed in that research. The dataset employed for this study presents a comprehensive collection of user movements, encompassing activities of daily living and various fall scenarios. Gathered within a controlled laboratory environment, the dataset consists of contributions from 17 individuals, comprising both male and female participants with an average age of  $30 \pm 8.02$  years, an average height of  $174.18 \pm 7.85$  cm, and an average weight of  $74.35 \pm 9.71$  kg. During data collection, users engaged in five distinct ADLs, including hitting the sensor, jumping, running and stopping, sitting on a chair, and pulling the sensor. Additionally, four different types of falls were meticulously simulated, encompassing forward falls, backward falls, falls on the right-hand side, and falls on the left-hand side. A trained model was employed to differentiate between fall events and ADLs, achieving this differentiation based on user movement features.

This approach facilitates a direct comparison of the proposed neural network with a Support Vector Machine algorithm. Accordingly, the training employs the dataset introduced in Ref. 62, where 17 volunteers executed 25 ADLs and 20 falls in a controlled setting, accumulating a total of 765 labeled movements documented with the MetaMotionR sensor. Conversely, the testing phase employed the dataset captured with the Puck.js sensor in Ref. 54, encompassing data from 11 volunteers who performed eight falls and six ADLs, with a total of 154 movements. Both datasets can be found elsewhere.<sup>e</sup> By training and testing in different datasets, we also evaluate the generality of the models under comparison across sensors. The findings of this study, alongside the results from Ref. 54, are delineated in Table 1. The analysis of the outcomes yields

that the solution outlined in this work outperforms the one presented in Ref. 54, achieving an accuracy of 95.75%, a specificity of 99.38%, and a sensitivity of 93.07%.

The second experiment undertaken for the evaluation of the system resorts to the SisFall dataset,<sup>55</sup> a dataset commonly referenced in current state-of-the-art studies. Despite the existence of a large number of datasets designed for collecting information to differentiate between ADLs and falls, virtually none of them gather specific data about the target users, i.e. the elderly. Some of the most relevant datasets in recent years include SisFall, UniMiB SHAR, UMAFall, Graz, and Gravity Project. In all these datasets, the majority of participants are in the age range of 20–29 years. The only dataset that includes older users is SisFall, making it one of the most widely used datasets in various studies and models related to fall detection.<sup>66</sup> This dataset compiles falls and ADLs executed by 38 different individuals, segmented into two groups: young adults and older adults, thereby offering a substantial sample across varied age groups. It should be noted that, with the exception of one individual from the older adult group, all fall simulations were carried out by the young adult participants. To facilitate the training of the model presented in this study with the SisFall dataset, and taking advantage of its extensive data pool, a 10-fold cross-validation approach was adopted. This strategy involved dividing the dataset into 10 segments, and training the model 10 times, each time excluding a different segment from the training phase and using it for testing instead. The average results from this cross-validation can be viewed in Table 2, alongside the results from other studies that employed the same dataset. In that table the model used is the full version.

The data indicates that the full model presented in this study demonstrates a performance comparable or superior to other deep learning models

<sup>e</sup><https://www.kaggle.com/jesusfrui/z/datasets/fall-dataset>.

Table 2. Comparison of different models using SisFall dataset.<sup>55</sup>

Work	Accuracy	Specificity	Sensitivity	Location	Algorithm	Parameters
Our full approach	<b>98.67%</b>	98.08%	98.58%	Waist	Self-Attention	<b>3347</b>
Ref. 69	97.46%	96.91%	98.04%	Waist	CNN	244,1702
Ref. 6	98.33%	97.93%	98.73%	Waist	LSTM	5,606,400
Ref. 49	98.61%	<b>99.80%</b>	<b>98.62%</b>	Waist	CNN	875,456

referenced in academic literature. Despite the similar performance metrics, the proposed model holds a distinctive advantage due to its considerably lower number of parameters — only 0.38% of the parameter size of the closest approach — resulting in a less complex model that retains efficiency, making it suitable for deployment at the edge.

The last experiment consists in conducting a comprehensive assessment using all three datasets employed in the previous evaluations. This experiment is intended to evaluate the model’s adaptability to heterogeneous data, given that the three datasets were acquired through various devices.

To facilitate this evaluation, initially, 201 samples from the SisFall dataset and another 198 from the datasets referenced in Refs. 62 and 54 (167 and 31 measures, respectively) were isolated, thereby comprising 399 samples for testing (each sample representing a 1-s window of the user’s inertial measurements). The remaining data were employed to train the model.

The outcomes, corresponding to both the full and lightweight models, are presented in Table 3. These results are slightly inferior to those achieved with the SisFall dataset alone, potentially indicating the model’s difficulty in accommodating the heterogeneity present across all the data. Yet, the results are still substantial, achieving an accuracy of 95.29%, a specificity of 93.68%, and a sensitivity of 96.66%. A closer inspection of these figures reveals that the model’s primary challenge lies in accurately classifying certain ADLs, as evidenced by the resulting specificity score. The lowest percentage in specificity

highlights an issue, as it implies that the system has some false negatives. These occurrences can lead to serious problems since they represent instances of falls that go undetected and, consequently, may go unattended. Currently, the system tests are conducted in a controlled environment, with caregivers closely monitoring users. For the future and for a system intended for more independent use, this aspect needs improvement. While the lightweight model exhibits a marginally reduced performance, it maintains a close approximation to the full model, demonstrating an accuracy of 93.94%, a specificity of 89.99%, and a sensitivity of 97.29%. We consider this slight reduction worthy since it gives the model the ability to be hosted and run on the edge, rather than on the cloud or on a bigger device that will require communication.

Finally, for a comprehensive evaluation of the model concerning computational resources and inference times, a comparative analysis was conducted against models that provided such metrics in their respective studies. Our lightweight model, as outlined in Table 4, demonstrates superior results in terms of inference time, closely resembling the performance of models presented in Refs. 67 and 68. Notably, our model exhibits significantly reduced memory usage compared to those studies reporting such metrics. In terms of overall performance metrics, our system consistently outperforms others. Turning attention specifically to the presented model, it shows slightly lower accuracy compared to the models presented here, with differences typically not exceeding 3%.

Table 3. Results of our model trained and tested on a combination of SisFall<sup>55</sup> as well as Refs. 62 and 54.

Work	Accuracy	Specificity	Sensitivity	Location	Algorithm
Our full approach	95.29%	93.68%	96.66%	Waist	Self-Attention
Our lite approach	93.94%	89.99%	97.29%	Waist	Self-Attention



Table 4. Comparison of various models based on inference time and memory usage.

Model	Inference time (ms)	Memory (kB)	Accuracy	Specificity	Sensitivity
Our lite approach	15	6313	93.94%	89.99%	97.29%
Ref. 70	37	61,084	95.55%	94.86%	95.1%
Ref. 71	99	—	94.4%	95.7%	93%
Ref. 67	18	200,000	91.1%	—	—
Ref. 68	20	—	97%	—	—

#### 4.2. Energy consumption

It was aforementioned that the ambition of this work is to deliver a solution that is both versatile and user-friendly. In pursuit of this goal, a system was designed capable of operating in diverse settings without the need for significant modifications. Ensuring this adaptability, especially considering the majority of the software operate at the edge, relies significantly on the battery autonomy of the sensor hosting the neural network. Insufficient sensor autonomy could undermine the system’s reliability, unable to assure extended continuous user monitoring, and might even fail to detect a fall. Furthermore, the frequent charging requirement could pose an inconvenience for the user, potentially cultivating a negative perception of the solution and encouraging its abandonment. It is therefore essential to examine the device’s consumption at the edge to understand its viability for daily use.

The device is equipped with a battery boasting a capacity of 400 mAh. Throughout the predominant sampling phase of the sensor’s operation, it averages a consumption rate of about 16 mA when no inference is being executed. Based on these figures, it can be projected that the device can operate continuously for around 25 h.

Given the aforementioned results, the device has yet to achieve the autonomy necessary to provide users with extended service. It is therefore necessary to further investigate on the aspects of the firmware contribution to this constant consumption. Furthermore, a more thorough examination of the device is suggested to explore the feasibility of implementing the sensor’s TBA in a sleep mode, wherein most of its functionalities are turned off. This would ensure minimal power usage when no inference execution is required, thereby promoting energy efficiency.

#### 5. Conclusions and Future Work

In this study, a fully functional fall detection system has been introduced, designed for everyday use by individuals and adaptable to any setting, from unconstrained indoor environments like homes to outdoor areas, facilitated by the communication between the sensor and the Android device. The sensor’s compact size allows for easy placement on the user’s waist, making it comfortable and unobtrusive. In addition to its usability flexibility, a lightweight Transformer model on the edge has been developed with an accuracy of 95.29%, a specificity of 93.68%, and a sensitivity of 96.66%, matching the outcomes of more complex models found in current research but with a very small number of parameters. Consequently, a highly reliable fall detection system has been established, user-friendly in design, and offering significant versatility in its application.

It is important to note the balance achieved between the reduction in parameter size and the preservation of detection accuracy in our novel lightweight Transformer architecture. While the proposed system has successfully achieved a significant reduction in parameter size by 99.6%, leading to enhanced computational efficiency and a smaller memory footprint, it is necessary to acknowledge a minor compromise in detection accuracy. This slight decrease in accuracy, however, is counterbalanced by the substantial gains in computational efficiency and the architecture’s suitability for edge computing devices. It is believed this trade-off is a worthwhile consideration, particularly for the intended application of our system in resource-constrained environments. The revised architecture thus represents a strategic balance, optimizing for efficiency and practicality while maintaining a high level of accuracy, making it an effective solution for real-time fall detection in wearable devices.

In future work, there is an expectation to enhance and expand the system, not only for fall detection but also for fall prevention. This involves utilizing data from the “flight phase” to create a more comprehensive system that could be employed in the development of automatic airbags, thereby minimizing the consequences of a fall. Furthermore, in line with established methodologies in fall detection research, this work employs a three-dimensional SVM approach, incorporating the  $x$ -,  $y$ -, and  $z$ -axes. This multi-dimensional analysis is useful for capturing the varied nature of falls, which often involve complex movements beyond simple vertical descent. Nonetheless, further research is needed to determine the potential benefits of optimizing this approach by understanding the predominance of vertical movements in fall incidents, especially within the employed datasets. This exploration may lead to a more focused analysis on the vertical axis, potentially enhancing the efficiency and accuracy of our fall detection algorithm.

The system exhibits certain issues concerning consumption. For instance, the estimated battery life of the Nicla Sense ME sensor varies around 20 h, leading to a significant reduction in autonomy. Sensor consumption can be mitigated by optimizing firmware programming. A multitude of external libraries designed for Arduino, present in the firmware’s code, unnecessarily consume resources and contribute to increased energy usage. This adoption of external libraries not only results in higher consumption but also inflates the firmware’s size. This size reduction could be achieved by employing only the essential code, which could then be allocated to enhancing the completeness of the neural network. Further optimization on the use of TensorFlow Lite could be also explored. The aforementioned points will be addressed in future work.


### Acknowledgments


This paper is funded by MCIN/AEI/10.13039/501100011033 under the Grant TALENT-BELIEF (PID2020-116417RB-C44) and the Project MIR-ATAR TED2021-132149B-C41 funded by MCIN/AEI/10.13039/501100011033 and by the European Union NextGenerationEU/PRTR. Furthermore, Jesús Fernández-Bermejo has received funding from


the European Union, specifically through the European Social Fund Plus (ESF+) and the mobility support from the Vice-Rectorate for Scientific Policy of the University of Castilla-La Mancha.


### ORCID


Jesús Fernández-Bermejo  <https://orcid.org/0000-0002-4177-4526>

Jesús Martínez-del-Rincon  <https://orcid.org/0000-0002-9574-4138>

Javier Dorado  <https://orcid.org/0000-0003-4700-5557>

Xavier del Toro  <https://orcid.org/0000-0002-9996-8008>

María J. Santofimia  <https://orcid.org/0000-0001-6132-1199>

Juan C. Lopez  <https://orcid.org/0000-0002-7372-1568>

### References

1. A. Cook, R. Swindall, K. Spencer, C. Wadle, S. A. Cage, Y. Mohiuddin, M. Desai and S. Norwood, Hospitalization and readmission after single-level fall: A population-based sample, *Inj. Epidemiol.* **10**(1) (2023) 49.
2. X. Wang, J. Ellul and G. Azzopardi, Elderly fall detection systems: A literature survey, *Front. Robot. AI* **7** (2020) 71.
3. N. M. Catikkas, T. O. Erdogan, J. Y. Reginster, M. M. Oren, C. Kilic, M. A. Karan and G. Bahat, Prevalence and determinants of falls in community-dwelling older adults: A population-based cross-sectional study, *Clin. Nutr. ESPEN* **54** (2023) 688–689.
4. T. Karpusenko, M. Alfonsi, N. T. de Oliveira Cirino, E. Y. Ishigaki, A. Sanudo, S. M. P. Paschoal, L. E. G. Leme and M. R. Perracini, Factors associated with unrecovered falls among older adults, *Geriatr. Nurs.* **51** (2023) 323–329.
5. S. N. Robinovitch, F. Feldman, Y. Yang, R. Schonop, P. M. Leung, T. Sarraf, J. Sims-Gould and M. Loughin, Video capture of the circumstances of falls in elderly people residing in long-term care: An observational study, *Lancet* **381** (2013) 47–54.
6. E. Torti, A. Fontanella, M. Musci, N. Blago, D. Pau, F. Leporati and M. Piastra, Embedding recurrent neural networks in wearable systems for real-time fall detection, *Microprocess. Microsystems.* **71** (2019) 102895.
7. L. Thiamwong and J. Suwanno, Fear of falling and related factors in a community-based study of people 60 years and older in Thailand, *Int. J. Gerontol.* **11**(2) (2017) 80–84.

8. F. Mahmud and N. S. Sirat, Evaluation of three-axial wireless-based accelerometer for fall detection analysis, *Int. J. Integr. Eng.* **7**(3) (2015) 15–20.
9. H. Djelouat, H. Baali, A. Amira and F. Bensaali, CS-based fall detection for connected health applications, in *Proc. 2017 Fourth Int. Conf. Advances in Biomedical Engineering (ICABME)* (IEEE, 2017), pp. 1–4.
10. A. Mehmood, A. Nadeem, M. Ashraf, T. Alghamdi and M. S. Siddiqui, A novel fall detection algorithm for elderly using shimmer wearable sensors, *Health Technol.* **9** (2019) 631–646.
11. K. Adhikari, H. Bouchachia and H. Nait-Charif, Activity recognition for indoor fall detection using convolutional neural network, in *Proc. 2017 Fifteenth IAPR Int. Conf. Machine Vision Applications (MVA)* (IEEE, 2017), pp. 81–84.
12. Z. Wang, V. Ramamoorthy, U. Gal and A. Guez, Possible life saver: A review on human fall detection technology, *Robotics* **9**(3) (2020) 55.
13. A. Singh, S. U. Rehman, S. Yongchareon and P. H. J. Chong, Sensor technologies for fall detection systems: A review, *IEEE Sens. J.* **20**(13) (2020) 6889–6919.
14. G. Şengül, M. Karakaya, S. Misra, O. O. Abayomi-Alli and R. Damaševičius, Deep learning based fall detection using smartwatches for healthcare applications, *Biomed. Signal Process. Control* **71** (2022) 103242.
15. R. Igual, C. Medrano and I. Plaza, Challenges, issues and trends in fall detection systems, *Biomed. Eng. Online* **12**(1) (2013) 66.
16. O. Darbin, C. Gubler, D. Naritoku, D. Dees, A. Martino and E. Adams, Parkinsonian balance deficits quantified using a game industry board and a specific battery of four paradigms, *Front. Hum. Neurosci.* **10** (2016) 431.
17. L. Ren and Y. Peng, Research of fall detection and fall prevention technologies: A systematic review, *IEEE Access* **7** (2019) 77702–77722.
18. J.-S. Lee and H.-H. Tseng, Development of an enhanced threshold-based fall detection system using smartphones with built-in accelerometers, *IEEE Sens. J.* **19**(18) (2019) 8293–8302.
19. A. Rezaei, A. Mascheroni, M. C. Stevens, R. Argha, M. Papandrea, A. Puiatti and N. H. Lovell, Unobtrusive human fall detection system using mmWave radar and data driven methods, *IEEE Sens. J.* **23**(7) (2023) 7968–7976.
20. H. Fu and J. Gao, Human fall detection based on posture estimation and infrared thermography, *IEEE Sens. J.* **23**(20) (2023) 24744–24751.
21. D. P. Sangeetha, S. Vijayalakshmi, S. Arunachalam, K. Kokila, U. Prakash and S. Swetha, Fall detection for elderly people using video-based analysis, *J. Adv. Res. Dyn. Control Syst.* **12** (2020) 232–239.
22. D.-W. Lee, K. Jun, K. Naheem and M. S. Kim, Deep neural network-based double-check method for fall detection using IMU-L sensor and RGB camera data, *IEEE Access* **9** (2021) 48064–48079.
23. S. Maldonado-Bascón, C. Iglesias-Iglesias, P. Martín-Martín and S. Lafuente-Arroyo, Fallen people detection capabilities using assistive robot, *Electronics* **8**(9) (2019) 915.
24. A. Y. Alaoui, S. El Fkihi and R. O. H. Thami, Fall detection for elderly people using the variation of key points of human skeleton, *IEEE Access* **7** (2019) 154786–154795.
25. M. Brenner, N. H. Reyes, T. Susnjak and A. L. C. Barczak, RGB-D and thermal sensor fusion: A systematic literature review, *IEEE Access* **11** (2023) 82410–82442.
26. C.-B. Lin, Z. Dong, W.-K. Kuan and Y.-F. Huang, A framework for fall detection based on OpenPose skeleton and LSTM/GRU models, *Appl. Sci.* **11**(1) (2021) 329.
27. D. Zhao, J. Yang, M. O. Okoye and S. Wang, Walking assist robot: A novel non-contact abnormal gait recognition approach based on extended set membership filter, *IEEE Access* **7** (2019) 76741–76753.
28. A. Naser, A. Lotfi, M. D. Mwanje and J. Zhong, Privacy-preserving, thermal vision with human in the loop fall detection alert system, *IEEE Trans. Hum.-Mach. Syst.* **53**(1) (2023) 164–175.
29. F. Riquelme, C. Espinoza, T. Rodenas, J.-G. Minonzio and C. Taramasco, eHomeSeniors dataset: An infrared thermal sensor dataset for automatic fall detection research, *Sensors* **19**(20) (2019) 4565.
30. F. Luna-Perejón, L. Muñoz-Saavedra, J. Civit-Masot, A. Civit and M. Domínguez-Morales, AnkFall — Falls, falling risks and daily-life activities dataset with an ankle-placed accelerometer and training using recurrent neural networks, *Sensors* **21**(5) (2021) 1889.
31. X. Xi, W. Jiang, Z. Lü, S. M. Miran and Z.-Z. Luo, Daily activity monitoring and fall detection based on surface electromyography and plantar pressure, *Complexity* **2020** (2020) 9532067.
32. Y. Zhou, R. Zia ur Rehman, C. Hansen, W. Maetzler, S. Del Din, L. Rochester, T. Hortobágyi and C. J. Lamoth, Classification of neurological patients to identify fallers based on spatial-temporal gait characteristics measured by a wearable device, *Sensors* **20**(15) (2020) 4098.
33. D. Mrozek, A. Koczur and B. Małysiak-Mrozek, Fall detection in older adults with mobile IoT devices and machine learning in the cloud and on the edge, *Inf. Sci.* **537** (2020) 132–147.
34. K. Nishio, T. Kaburagi, Y. Hamada, T. Matsumoto, S. Kumagai and Y. Kurihara, Construction of an aggregated fall detection model utilizing a microwave Doppler sensor, *IEEE Internet Things J.* **9**(3) (2021) 2044–2055.
35. G. Mokhtari, S. Aminikhanghahi, Q. Zhang and D. J. Cook, Fall detection in smart home environments

- using UWB sensors and unsupervised change detection, *J. Reliab. Intell. Environ.* **4**(3) (2018) 131–139.
36. J.-K. Kim, K. Lee and S. G. Hong, Detection of important features and comparison of datasets for fall detection based on wrist-wearable devices, *Expert Syst. Appl.* **234** (2023) 121034.
  37. L. Montanini, A. Del Campo, D. Perla, S. Spinsante and E. Gambi, A footwear-based methodology for fall detection, *IEEE Sens. J.* **18**(3) (2017) 1233–1242.
  38. M. M. Hassan, A. Gumaei, G. Aloï, G. Fortino and M. Zhou, A smartphone-enabled fall detection framework for elderly people in connected home healthcare, *IEEE Netw.* **33**(6) (2019) 58–63.
  39. K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink and J. Schmidhuber, LSTM: A search space odyssey, *IEEE Trans. Neural Netw. Learn. Syst.* **28**(10) (2016) 2222–2232.
  40. C.-C. Lin, C.-Y. Yang, Z. Zhou and S. Wu, Intelligent health monitoring system based on smart clothing, *Int. J. Distrib. Sens. Netw.* **14**(8) (2018) 1550147718794318.
  41. C. Taramasco, T. Rodenas, F. Martinez, P. Fuentes, R. Munoz, R. Olivares, V. H. C. De Albuquerque and J. Demongeot, A novel monitoring system for fall detection in older people, *IEEE Access* **6** (2018) 43563–43574.
  42. A. H. Ngu, V. Metsis, S. Coyne, P. Srinivas, T. Salad, U. Mahmud and K. H. Chee, Personalized watch-based fall detection using a collaborative edge-cloud framework, *Int. J. Neural Syst.* **32**(12) (2022) 2250048.
  43. A. Bhattacharya, T. Baweja and S. Karri, Epileptic seizure prediction using deep transformer model, *Int. J. Neural Syst.* **32**(02) (2022) 2150058.
  44. J. Lian and F. Xu, Epileptic EEG classification via graph transformer network, *Int. J. Neural Syst.* **33**(8) (2023) 2350042.
  45. W. Y. Peh, P. Thangavel, Y. Yao, J. Thomas, Y.-L. Tan and J. Dauwels, Six-center assessment of CNN-Transformer with belief matching loss for patient-independent seizure detection in EEG, *Int. J. Neural Syst.* **33**(03) (2023) 2350012.
  46. H. S. Nogay and H. Adeli, Machine learning (ML) for the diagnosis of autism spectrum disorder (ASD) using brain imaging, *Rev. Neurosci.* **31**(8) (2020) 825–841.
  47. H. S. Nogay and H. Adeli, Detection of epileptic seizure using pretrained deep convolutional neural network and transfer learning, *Eur. Neurol.* **83**(6) (2021) 602–614.
  48. N. Zerrouki and A. Houacine, Combined curvelets and hidden Markov models for human fall detection, *Multimed. Tools Appl.* **77** (2018) 6405–6424.
  49. J. He, Z. Zhang, X. Wang and S. Yang, A low power fall sensing technology based on FD-CNN, *IEEE Sens. J.* **19**(13) (2019) 5110–5118.
  50. D. Ajerla, S. Mahfuz and F. Zulkernine, A real-time patient monitoring framework for fall detection, *Wirel. Commun. Mob. Comput.* **2019** (2019) 9507938.
  51. M. Musci, D. De Martini, N. Blago, T. Facchinetti and M. Piastra, Online fall detection using recurrent neural networks on smart wearable devices, *IEEE Trans. Emerg. Top. Comput.* **9**(3) (2021) 1276–1289.
  52. M. Farsi, Application of ensemble RNN deep neural network to the fall detection through IoT environment, *Alex. Eng. J.* **60**(1) (2021) 199–211.
  53. J. Ding and Y. Wang, A WiFi-based smart home fall detection system using recurrent neural network, *IEEE Trans. Consum. Electron.* **66**(4) (2020) 308–317.
  54. J. F. B. Ruiz, J. D. Chaparro, M. J. S. Romero, F. J. V. Molina, X. D. T. García, C. B. Peño, H. L. Solano, S. Colantonio, F. Flórez-Revuelta and J. C. López, Bedtime monitoring for fall detection and prevention in older adults, *Int. J. Environ. Res. Public Health* **19**(12) (2022) 7139.
  55. A. Sucerquia, J. D. López and J. F. Vargas-Bonilla, SisFall: A fall and movement dataset, *Sensors (Switzerland)* **17**(1) (2017) 198.
  56. A. T. Özdemir, An analysis on sensor locations of the human body for wearable fall detection devices: Principles and practice, *Sensors* **16**(8) (2016) 1161.
  57. N. S. Suriani, F. A. N. Rashid and N. Y. Yunos, Optimal accelerometer placement for fall detection of rehabilitation patients, *J. Telecommun. Electron. Comput. Eng.* **10**(2–5) (2018) 25–29.
  58. H. W. Guo, Y. T. Hsieh, Y. S. Huang, J. C. Chien, K. Haraikawa and J. S. Shieh, A threshold-based algorithm of fall detection using a wearable device with tri-axial accelerometer and gyroscope, in *Proc. 2015 Int. Conf. Intelligent Informatics and Biomedical Sciences (ICIIBMS 2015)* (IEEE, 2016), pp. 54–57.
  59. Y. Hwang, W. Choi, H. An and C. G. Lee, Fall decision algorithm based on changeable velocity SVM threshold, in *Proc. 2019 IEEE Int. Conf. Consumer Electronics — Asia (ICCE-Asia 2019)* (IEEE, 2019), pp. 41–42.
  60. S. Abbate, M. Avvenuti, F. Bonatesta, G. Cola, P. Corsini and A. Vecchio, A smartphone-based fall detection system, *Pervasive Mob. Comput.* **8**(6) (2012) 883–899.
  61. M. Tarabini, B. Saggin, M. Bocciolone, D. Scaccabarozzi and M. Magni, Falls in older adults: Kinematic analyses with a crash test dummy, in *Proc. 2016 IEEE Int. Symp. Medical Measurements and Applications (MeMeA)* (IEEE, 2016), pp. 1–6.
  62. J. F. B. Ruiz, J. D. Chaparro, C. B. Peño, H. A. Llumiguano Solano, X. del Toro García and J. C. López López, A low-cost and unobtrusive system for fall detection, *Procedia Comput. Sci.* **192** (2021) 2160–2169.

63. P. Tsinganos and A. Skodras, A smartphone-based fall detection system for the elderly, in *Proc. 10th Int. Symp. Image and Signal Processing and Analysis (ISPA)* (IEEE, 2017), pp. 53–58.
64. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin, Attention is all you need, in *Advances in Neural Information Processing Systems*, Vol. 30 (Curran Associates, Inc., 2017), pp. 5999–6009.
65. G. Wang, Q. Li, L. Wang, Y. Zhang and Z. Liu, Elderly fall detection with an accelerometer using lightweight neural networks, *Electronics (Switzerland)* **8**(11) (2019) 1354.
66. E. Casilari, J.-A. Santoyo-Ramón and J.-M. Cano-García, Analysis of public datasets for wearable fall detection systems, *Sensors* **17**(7) (2017) 1513.
67. K. Kim, G. Yun, S.-K. Park and D. H. Kim, Efficient fall detection for a healthcare robot system based on 3-axis accelerometer and depth sensor fusion with LSTM networks, in *Proc. 2022 26th Int. Conf. Pattern Recognition (ICPR)* (IEEE, 2022), pp. 2207–2212.
68. E. A. Kamiński, P. Bonato and H. H. Asada, Time-critical fall prediction based on Lipschitz data analysis and design of a reconfigurable walker for preventing fall injuries, *IEEE Access* **12** (2024) 1822–1838.
69. T. Xu, H. Se and J. Liu, A fusion fall detection algorithm combining threshold-based method and convolutional neural network, *Microprocess. Microsyst.* **82** (2021) 103828.
70. O. Z. Salah, S. K. Selvaperumal and R. Abdulla, Accelerometer-based elderly fall detection system using edge artificial intelligence architecture, *Int. J. Electr. Comput. Eng.* **12** (2022) 4430–4438.
71. I. Araújo, M. Pereira, W. Silva, I. Linhares, V. Marx, A. Andrade, R. Andrade and M. Castro, Machine learning and cloud enabled fall detection system using data from wearable devices: Deployment and evaluation, in *Proc. Anais do XXII Simpósio Brasileiro de Computação Aplicada à Saúde (SBCAS)* (2022), pp. 413–424.