UNIVERSITÀ
POLITECNICA
DELLE MARCHE

Faculty of Engineering
Master's Degree in Biomedical Engineering

Embedded Fall Detection System using Deep Learning

Candidate:
Alaa Alnasef

Supervisor:
Prof. Lorenzo Palma

Co-Supervisor:
Eng. Manuela Pigini

Academic Year 2022-2023

UNIVERSITÀ
POLITECNICA
DELLE MARCHE

Faculty of Engineering
Master's Degree in Biomedical Engineering

Embedded Fall Detection System using Deep Learning

Candidate:
Alaa Alnasef

Supervisor:
Prof. Lorenzo Palma

Co-Supervisor:
Eng. Manuela Pigini

Academic Year 2022-2023

Università Politecnica delle Marche
Faculty of Engineering
Master's Degree in Biomedical Engineering
Via Brecce Bianche – 60131 Ancona (AN), Italy

To my new beginning

Acknowledgments

I would like to extend my heartfelt appreciation to the individuals and organizations whose invaluable contributions were instrumental in the successful completion of my research and thesis:

First and foremost, I am profoundly grateful to Prof. Lorenzo Palma, my esteemed thesis advisor, whose unwavering guidance and expertise have been pivotal in shaping my research. I consider it an honor to have had the privilege of working under your mentorship.

My gratitude extends to the esse.ti team, with a special mention to my supervisor, Eng. Manuela Pigini, for their invaluable insights during my internship. Their guidance and technical support significantly influenced the direction of my research.

I also want to express my deep appreciation to Damascus University, particularly Prof. Hanan Mukhaiber and Prof. Hani Amasha, for their exceptional support. Their encouragement and assistance played a critical role in my acceptance at UNIVPM.

I owe a profound debt of gratitude to my family. To my father, Naser, my mother, Mohdia, my sister, Afraa, and lastly, my brother, Walaa, your unwavering belief in me has been a driving force in reaching the point I am today.
To my uncles Mohammed, Mhidi, Ahmad, Abdulghani, and Najie Alshamlan.
To my aunts Amal, Sanaa, and Yasmeen, and to my grandmother Sobhia.
To my friends, Saleh El Debuch, Taleb Kashour, Mohamad A-sharout, Muhammad Kalet, Mohammad Omki, Mohammad Najie, Hassan Alshamlan, and to the persons who were always there when I needed them Ayham Al-Taleb and Mhd Jafar Mortada.
To all my other friends whom I may have inadvertently omitted, I want to sincerely thank you as well. Your friendship has been invaluable to me.
To the CMS Service team, Reda Alkhateeb, Majd Khalifa, and Amer Darkazanly.
Lastly, I am thankful for the nurturing environment of my hometown, Hajeen, Syria, which has played a significant role in shaping my academic journey.

To all those mentioned, I deeply appreciate your contributions and unwavering support, which were integral to the successful completion of this research.

Ancona, Aug 2023

Alaa Alnasef

Abstract

The balance control system, crucial for maintaining stability and preventing falls, has garnered significant attention due to its decline in the elderly and individuals with various pathologies. As the global aging population continues to grow and life expectancy increases, preserving mobility has become increasingly vital. This thesis addresses the pressing need for an accurate, portable, and affordable device for fall detection.

The World Health Organization reports alarming statistics, with a substantial percentage of individuals over 65 experiencing falls annually, often leading to severe consequences. Traditional balance assessment methods rely on costly and immobile dynamometric force platforms, limiting their practicality for widespread use.

Fall detection systems can help to mitigate this risk by alerting caregivers when a fall has occurred. However, many existing fall detection systems are expensive, bulky, and/or require specialized installation.

Recent years have witnessed a remarkable shift towards body-wearable sensor technology, offering an innovative approach to assessing an individual's motion and activity. These sensors, including accelerometers and gyroscopes, enable cost-effective, portable, and versatile solutions for measuring three-dimensional movements. This development aligns with the increasing importance of fall detection as falls become a growing public health concern.

In this thesis, we propose a fall detection system that uses a neural network to classify data from a tri-axial accelerometer tri-axial gyroscope, and a pressure sensor. Our system is designed to be low-cost, portable, and easy to install.

The aim of this study is to develop a high-accuracy model that is small enough to run on an embedded controller and also to improve the model to work with different datasets. Lastly, assess the benefit of adding a pressure signal to the dataset.

We evaluated our system on two datasets the SisFall dataset, and a self-collected dataset. A combined dataset that includes data from SisFall and self-collected datasets was also assessed. Moreover, the self-collected dataset contains in addition to a pressure signal a new fall type which was not introduced in the SisFall dataset which is Syncope. Our system achieved a testing accuracy of 99.38% on the combined dataset, demonstrating its potential for use in a real-world setting. moreover, adding the pressure signal to the training data led to improving the accuracy slightly and lowered the false positive and false negative when compared to the results of training the model without the inclusion of the pressure signal.

Our system has the potential to improve the safety and independence of individuals

of all ages, including elderly individuals and workers, by ensuring immediate assistance is provided when a fall incident happens. This rapid response not only reduces the time between the incident and help arriving but also significantly diminishes the potential consequences of the fall.

Contents

List of Figures

List of Tables

Chapter 1

Balance System in Humans

1.1 Introduction

Researchers and clinicians have been compelled to gain a deeper understanding of the balance control system due to its deterioration in the elderly and various pathologies. This necessity arises from the growing aging population and increased life expectancy, making the preservation of mobility increasingly vital. Consequently, efforts are being made to comprehend the functioning of the balance control system and establish methods to assess its condition accurately [1].

There are three primary sensory systems crucial for maintaining balance and posture. Vision plays a key role in planning our movement and navigating around obstacles. The vestibular system acts as our internal "gyro," detecting linear and angular accelerations. The somatosensory system comprises numerous sensors that perceive the position and speed of each body segment, detect contact with external objects (including the ground), and determine the orientation of gravity [1].

The World Health Organization (WHO) reports that roughly 28% to 35% of those over 65 have a fall each year. When referring to older people above the age of 70, this ratio rises to 32–42%. According to projections, the number of people over 65 in the world will rise by 21.64% by the year 2050. Due to variables including diminished leg strength, long-term pharmaceutical side effects, eyesight impairments, and a general reduction in tissue strength, the impact and danger of falls tend to dramatically grow as individuals age. Falls can result in wounds that cause discomfort, disabling conditions, and in severe cases, early demise [2].

Traditionally, balance assessment has relied on the use of a dynamometric force platform, which is an instrument used to measure the forces exerted by a person on a support base. It enables the evaluation of postural sway by recording the displacement of the center of pressure, which represents the point where the ground reaction force is applied. While force platform-based assessments are considered the gold standard for measuring balance in both healthy and diseased individuals, they have limitations. These evaluations require clinical expertise and a specialized laboratory space. Additionally, they are costly, heavy, and not easily transportable, making them impractical for conducting balance assessments outside of the laboratory setting, such as in a patient's home, which may be necessary in certain clinical

scenarios where visiting a lab is challenging for the patient [3].

In recent years, there has been a significant development in the use of body-wearable sensor technology, which relies on electromechanical sensors, to accurately detect and monitor the body's movements and physical activity in unrestricted conditions. This technology offers a new approach to assessing an individual's motion and activity. For instance, the combination of multiple accelerometers and angular-rate sensors, such as gyroscopes, shows great potential in creating hybrid kinematic sensor modules capable of measuring the three-dimensional movements of various body segments. One of the key advantages of using body-wearable sensors is their affordability, making them a cost-effective option. Additionally, this technology does not necessitate a specific environment or the installation of any particular infrastructure, further enhancing its practicality and versatility [3].

This chapter will provide a global overview of the state of the art regarding human balance. In the following sections, we will delve into the balance system in healthy humans, exploring the three primary sensory systems involved in maintaining balance and posture: vision, the vestibular system, and the somatosensory system. We will then shift our focus to fall detection in the elderly, discussing the challenges and available methods in identifying falls. Additionally, we will examine statistics related to fall detection and injuries. Finally, we will explore traditional methods used to prevent falls before introducing smart and innovative approaches that have emerged in recent years.

1.2 Balance system in healthy humans

1.2.1 Introduction

The upright stance position is intrinsically unstable since even the slightest deviation from perfect alignment creates gravitational forces that cause the body to move towards the ground. Stability is achieved by generating appropriate torques in the joints to correct for deviations from the desired orientation, which are detected by sensory systems such as somatosensory/proprioceptive, visual, and vestibular systems. Balance control can be seen as a closed-loop feedback control system Figure 1.1, where integrating sensory orientation information is a crucial part of the overall system. However, the feedback nature of the system imposes limitations on the sensory integration process. Analyzing body sway resulting from balance disruptions enables the measurement of "sensory weights," representing the relative contributions of different sensory systems to estimate the internal orientation and generate corrective actions. Research demonstrates that sensory weights are not fixed, but vary depending on environmental conditions, experimental factors, and neurological disorders affecting the quality of sensory information from different systems. Rapid changes in environmental conditions necessitate swift sensory reweighting to prevent instability caused by insufficient or excessive corrective action [4].

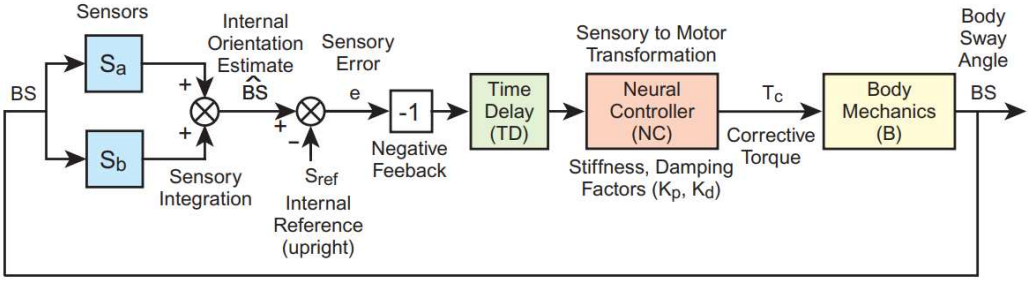

Figure 1.1: Block diagram showing interconnections of components of a closed-loop negative-feedback control system capable of maintaining a stable upright stance [4].

1.2.2 Control of balance and posture

The term "posture" refers to the body's position in space and serves the purpose of maintaining balance during both dynamic movements and stillness. Posture is influenced by various factors, including neurophysiological, biomechanical, and psychoemotional factors, which are connected to the evolutionary development of species. Posture is an automatic and unconscious state that represents the body's response to the gravitational force. It is sustained through the contraction of skeletal muscles, coordinated by a range of stimuli of different natures and continuous adjustments of neuromuscular nature. Therefore, posture can be defined as any position that ensures balance maintenance with optimal stability, minimal energy expenditure, and minimal stress on anatomical structures [5].

The postural control system serves two primary purposes: firstly, it establishes and maintains posture in response to gravity, ensuring balance; and secondly, it establishes and maintains the orientation and position of body segments as a reference for perception and action in relation to the external environment. These dual functions rely on four key components: reference values, which include the orientation of body segments and the position of the center of gravity (forming an internal representation known as the postural body scheme); multisensory inputs that regulate the orientation and stabilization of body segments; and adaptable postural reactions or anticipations that enable balance recovery following disturbances or postural stability during voluntary movement [6].

1.2.3 Functional Neuroanatomy for Posture and Gait Control

One important aspect of cortical control in posture is understanding how the brain obtains an internal representation of body motion and the vertical position. The perception of postural verticality relies on the integration of visual, somatosensory, and vestibular information. Among these sensory inputs, the vestibular system plays a crucial role due to its direct connection to gravity and its ability to provide the brain with information about three-dimensional head movements. Vestibular signals are

transmitted to the posterior thalamus through bilateral vestibulothalamic projections. Although there is no single cortical area exclusively dedicated to vestibular input, various regions in the cerebral cortex receive vestibular information, including the frontal eye field, PM, somatosensory cortex, ventral intraparietal cortex, medial superior temporal area, and parieto-insular vestibular cortex (PIVC) [7].

Postural dyscontrol can arise from various factors that impact the sensorimotor system, including pathological conditions that affect one or multiple components. Additionally, age-related changes can contribute to postural dyscontrol due to the natural decline in the efficiency of the vestibular, visual, and proprioceptive functions. As a result, neuromusculoskeletal disorders can lead to the deterioration of the postural control system [1].

The central nervous system combines sensory information and transmits neural signals to the muscles, leading to the generation of neuromuscular responses. The underlying concept is that postural stabilization relies on a feedforward mechanism, where control is executed through a series of anticipatory motor commands [8].

1.2.4 Mechanical control of balance and posture

The center of pressure (COP) Figure 1.2, which is the point where the resultant ground reaction force (GRF) is applied to a body, provides valuable information for assessing postural stability. This is because balance control aims to regulate the center of mass (COM) of the human body by adaptively changing the position of the COP. By using Newtonian mechanics to develop two equations that relate the two-dimensional COP coordinates to the GRF components, one can easily determine the location of the COP using a force plate. An important property of these two equations is that for a given COP position, there exists an infinite number of GRF component combinations that can satisfy these two equations [9].

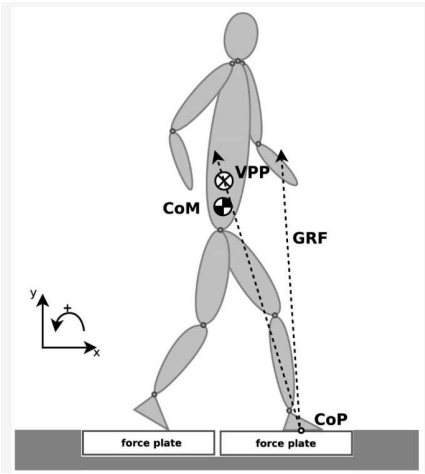

Figure 1.2: This Figure shows the center of mass (CoM), ground reaction force (GRF), and center of pressure (CoP). [10].

To maintain balance, the center of mass (COM) needs to stay within the boundaries of the support area. To achieve this, the postural control system must create stabilizing forces that adjust the position of the center of pressure (COP) in response to changes in the COM [1, 11].

A typical force plate Figure 1.3 has four force transducers positioned at the corners of the force plate. The coordinate system used to measure forces has three axes labeled as x, y, and z, which correspond to the medial-lateral, anterior-posterior, and vertical directions, respectively. The center of the force plate is located at the origin of the coordinate system. It's important to note that the arrows associated with each axis indicate the positive direction in which the corresponding axis points [9].

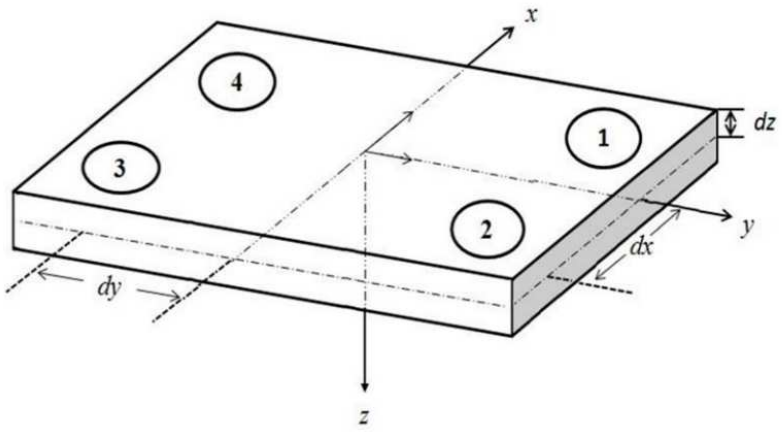

Figure 1.3: Force plate, its coordinate system, its three geometrical parameters, and locations of the four force sensors [9].

Assuming Fi represents the GRF component at the ith corner of the force plate, and fx, fy, and fz represent the components of the GRF in the medial-lateral, anterior-posterior, and vertical directions, respectively, the vector f can be used to represent the resultant GRF.

$$f = f_x i + f_y j + f_z k = F_1 + F_2 + F_3 + F_4 \tag{1.1}$$

The unit vectors i, j, and k correspond to the x, y, and z-axes, respectively. F_{iz} represents the vertical component of F_i and the vertical component of the resultant GRF f can be expressed using F_{iz}.

$$f_z = F_{1z} + F_{2z} + F_{3z} + F_{4z} \tag{1.2}$$

The components of Fi in the medial-lateral and anterior-posterior directions are represented by F_{ix} and F_{iy}, respectively. Using F_{ix} and F_{iy}, the corresponding medial-lateral and anterior-posterior components of the resultant GRF f can be determined using Eq.(1.3) and Eq.(1.4).

$$f_x = F_{1x} + F_{2x} + F_{3x} + F_{4x} \tag{1.3}$$

$$f_y = F_{1y} + F_{2y} + F_{3y} + F_{4y} \tag{1.4}$$

To calculate the coordinates of the COP, we need to have knowledge of three geometrical parameters of the force plate: dx, dy, and dz. These parameters are illustrated in Figure 1.3, where dx and dy represent the distances from the coordinate axes to the force sensors along the x-axis and y-axis, respectively, and dz represents the distance from the origin of the coordinate system to the support surface of the force plate along the z-axis. Once dx, dy, and dz are known, we can use Newtonian mechanics to derive equations Eq.(1.5) and Eq.(1.6) that determine the COP coordinates [9].

$$COP_x = \frac{d_x F_{xz} + d_z f_x}{f_z} \tag{1.5}$$

$$COP_y = \frac{d_y F_{yz} + d_z f_y}{f_z} \tag{1.6}$$

where:

$$f_{xz} = F_{1z} + F_{4z} - F_{2z} - F_{3z} \tag{1.7}$$

$$f_{yz} = F_{1z} + F_{2z} - F_{3z} - F_{4z} \tag{1.8}$$

Eq.(1.7) shows how the four vertical components of the GRF are combined to affect the movement of the COPx, which represents the medial-lateral direction coordinate of the COP. Likewise, Eq.(1.8) illustrates how the four vertical components of the GRF are combined to alter the movement of the COPy, which represents the anterior-posterior direction coordinate of the COP [9].

When standing still, the force of gravity acting on the body is much stronger than the force of inertia in the vertical direction. Therefore, it is assumed that the vertical component of the resultant GRF, f_z, is equal to the weight force acting on the body. This simplifies calculations, and when combined with the mass of the subject (M) and acceleration due to gravity (g), allows for the determination of other parameters related to postural stability. we have:

$$f_z = F_{1z} + F_{2z} + F_{3z} + F_{4z} \approx Mg \tag{1.9}$$

Under the assumption of a constant f_z, Eq.(1.5) and Eq.(1.6) can be rewritten as:

$$COP_x = c_{xz} F_{xz} + c_{zz} f_x \tag{1.10}$$

$$COP_y = c_{yz} F_{yz} + c_{zz} f_y \tag{1.11}$$

Where $c_{xz} = d_x/Mg$, $c_{zz} = d_z/Mg$, and $c_{yz} = d_y/Mg$. Eq.(1.10) shows that COPx is determined by its vertical GRF component F_{xz} and its medial-lateral horizontal

GRF component f_x. Similarly, Eq.(1.11) indicates that COPy is determined by its vertical GRF component F_{yz} and its anterior-posterior horizontal GRF component f_y.

1.2.5 Inverted pendulum model

The relationship between the resultant COP and the COM during the quiet stance can be analyzed using a bio-mechanical model of balance, commonly called an Inverted pendulum model. This model assumes that the human body can be treated as a single solid object connected to the ground through the ankle joint. By visualizing the body in a standing position as a pendulum swinging from the ankle, the control of posture is conceptualized as a feedback control system. In this system, the center of mass (COM) of the body is regarded as the output and a passive variable, while the center of pressure (COP) is viewed as the control variable [11, 12].

There are multiple techniques available to estimate the center of mass (COM) of an individual. One widely used method involves analyzing data from an accelerometer positioned on the sacrum of the person, which is generally considered the optimal location for tracking the COM. While this method can provide accurate outcomes when the individual is standing still or walking in a straight line, it might not be suitable for evaluating the COM when the person is swaying markedly or performing movements like reaching tasks. This technique relies on a simplified model that treats the body as a single inverted pendulum rotating around the ankle joint (assuming minimal movement at the hip joint). However, this assumption isn't always valid, particularly when the person exhibits substantial swaying motions [3].

Figure 1.4 illustrates a subject standing erect on a force plate and swaying back and forth. Each figure in the sequence depicts the evolving situation at five distinct points in time, capturing the changes in the subject's posture and movement over time.

At Time 1, the center of gravity (COG) of the body is positioned ahead of the center of pressure (COP), and the angular velocity (w) is assumed to be clockwise. The body's weight (W) is balanced by the vertical reaction force (R), creating a "parallelogram of forces" that acts at distances g and p from the ankle joint respectively. It's important to note that both W and R remain constant during quiet standing. In this scenario, considering the body as an inverted pendulum pivoting around the ankle, there is a counterclockwise moment equal to Rp and a clockwise moment equal to Wg acting on the body.

$$Rp - Wg = I\alpha \tag{1.12}$$

where:

- I is the moment of inertia of the total body about the ankle joint (kg·m^2)
- α is the angular acceleration of the inverted pendulum (r·s^{-2})

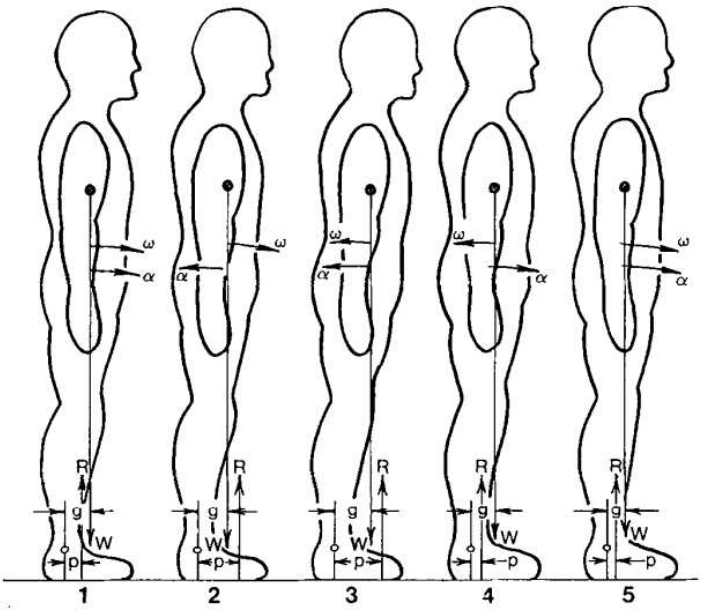

Figure 1.4: The subject is standing quietly on a force platform and swaying back and forth. We have five different points in the time described, each showing the locations of the center of gravity (g) and the center of pressure (p), as well as the associated angular accelerations (a) and angular velocities (w) [12].

If the product of Wg (W multiplied by g) is greater than Rp (R multiplied by p), the body will undergo a clockwise angular acceleration. To correct this forward sway, the subject will increase their center of pressure (COP) by activating the plantarflexion muscles, causing the COP to shift anteriorly relative to the center of gravity (COG) at Time 2. Now, Rp becomes greater than Wg. As a result, the angular acceleration (α) will reverse its direction, causing the angular velocity (w) to start decreasing. This reversal in w will continue until Time 3 when the time integral of α results in a complete reversal of the angular velocity, effectively changing its direction.

At this point, both the angular velocity (w) and angular acceleration (α) are counterclockwise, indicating that the body is experiencing a backward sway. When the central nervous system (CNS) detects this posterior shift of the center of gravity (COG) and determines that it needs correction, the subject decreases plantar flexor activation, causing the center of pressure (COP) to decrease until it lies posterior to the COG. Consequently,α (the angular acceleration) reverses its direction and becomes clockwise again at Time 4. Following a period, the angular velocity (w) will decrease and reverse, ultimately returning the body to its original conditions, as observed at Time 5.

From the sequence of conditions observed in the center of gravity (COG) and center of pressure (COP), it becomes evident that the plantar flexors and dorsiflexors, responsible for controlling the net ankle moment, play a role in regulating the

body's COG. However, it is apparent that the dynamic range of the COP needs to be somewhat greater than that of the COG. The COP must continuously move anteriorly and posteriorly in relation to the COG. If the COG were allowed to move within a few centimeters of the toes, it is possible that a corrective movement of the COP to the extreme ends of the toes would not be sufficient to reverse the angular velocity (w). In such a scenario, the subject would need to move a limb forward to prevent the forward fall [1].

The difference between the center of mass (COM) and the center of pressure (COP) is proportional to the acceleration of the COM [1].

$$COM - COP = KC\ddot{O}M \tag{1.13}$$

1.3 Approaches and principles of fall detection for elderly

The field of assistive technology for elderly individuals and patients has gained significant attention in research. This is driven by the healthcare industry's increasing need for such products and technology. As the global population of elderly individuals grows rapidly, there is a corresponding rise in demand for healthcare systems. Simultaneously, advancements in sensor, camera, and computer technologies have made it possible to develop these assistive technologies [13].

Assistive technology plays a crucial role in enhancing the independence of elderly individuals and patients while alleviating the strain caused by nursing shortages. One such example is the fall detector, which contributes to increased safety and care. Consequently, it boosts the self-assurance of the elderly and patients in maintaining an independent lifestyle.

1.3.1 Specification of characteristic of FALL

In this section, we not only recognize different types of falls but also define the unique attributes associated with each type. It is essential to identify various types of falls individually since we may need to address them differently. Additionally, specifying the characteristics of falls serves as a valuable tool. It aids us in comprehending existing algorithms and guides the development of new fall detection algorithms. This is important because algorithms must be designed based on specific fall characteristics.

Falls can be categorized into four types based on the circumstances in which they occur: falls while sleeping (from bed), falls while sitting (from a chair), falls while walking or standing on the floor, and falls while standing on supports like ladders or tools. Falls from standing on support primarily happen among working individuals, although they occasionally occur among the elderly during household activities. The elderly and patients are primarily at risk from the first three types of falls [13].

The characteristics of fall from sleeping (or bed)

1. A fall is a process lasting 1 to 3 seconds, consisting of several sub-actions.
2. The person is lying in bed at the beginning of the fall.
3. The body reduces its height from the bed height to the lying height (lying on the floor). Within the portion of this period, the body will fall in a free fall manner.
4. The lying body on the floor is near the bed.

The characteristics of fall from sitting (or chair)

1. A fall is a process lasting 1 to 3 seconds, consisting of several sub-actions.
2. The person is sitting in the chair at the beginning of the fall.
3. The head reduces its height from the sitting height to the lying height (lying on the floor). During this period, the head will fall in a free-fall manner
4. The lying body on the floor is near the chair.

The characteristics of fall from working or standing

1. A fall is a process lasting 1 to 2 seconds, consisting of several sub-actions.
2. The person stands at the beginning of a fall. Here we define a fall as from standing to lying on the floor. Normal people may say that fall does not include standing.
3. The head lies on the floor at the end of the fall process. The head would lie on the floor motionless or with little motion for a while.
4. A person falls roughly in one direction. As a result, both the head and the weight center of the person move approximately in one plane during falling.
5. The head reduces its height from the standing height to the lying height (lying on the floor). Within the portion of this period, the head will fall in a free fall manner.
6. The lying head is within a circle centered at the foot position of the last standing time and with the radium of the height of the person.

It is important to note that certain characteristics of falls can also be observed in normal activities. For example, a crouch involves a rapid decrease in head height, similar to a fall. Falls from a bed or chair tend to last longer due to the partial support provided by the furniture. To establish more precise time limits for each type of fall, we can gather additional statistical data [13].

1.3.2 Class hierarchy of Fall detection methods

Extensive efforts have been dedicated to the development of fall detection technology, driven by the significant demand, potential market, and societal value of such products. In recent years, a range of technologies has been successfully developed. Based on the approach used for fall detection, these technologies can be classified into three categories: wearable devices, ambiance devices, and camera-based (or vision-based) methods. Wearable devices can be further categorized into posture devices and motion devices. Ambiance devices can be divided into presence devices and posture devices. Camera-based methods are classified into three groups based on the principles employed: inactivity detection, 2D body shape change analysis, and 3D head motion analysis. The hierarchical classification of fall detection methods is illustrated in Figure 1.5.

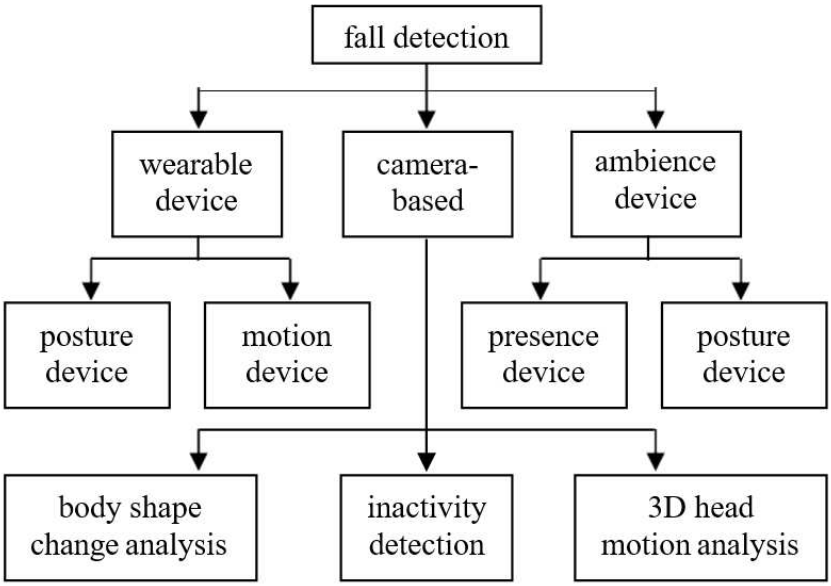

Figure 1.5: The hierarchy of approaches and classes of fall detection methods for elderly and patients [13].

While various approaches and methods exist for detecting falls in the elderly and patients, there is a common framework shared by existing fall detection devices and systems, as shown in Figure 1.6. The differentiation among these devices and systems lies in the complexity of each component. For instance, the data acquisition stage can range from a single simple sensor measuring one parameter to a combination of multiple sensors and cameras working together to collect signal and video data. Fall detection methods can also differ, ranging from a simple comparison of a sensed parameter with a threshold to a sophisticated image processing algorithm involving background subtraction, shape detection, and analysis of shape changes.

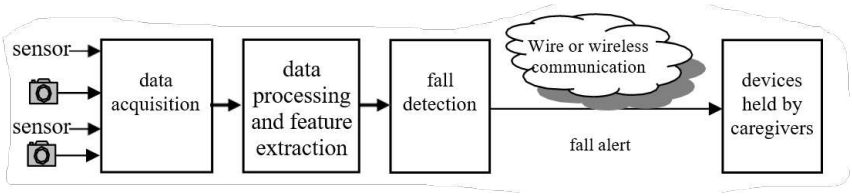

Figure 1.6: General framework of fall detection and alert device and system for elderly and patient [13].

1.4 Statistics about falls among elderly

Falls are a frequent occurrence and have significant financial implications, especially among individuals aged 65 and above in the United States. However, it is important to note that falls can be prevented and should not be considered an unavoidable consequence of aging.

In the U.S., older adults aged 65 and above experience a fall every second of every day, making falls the primary cause of injury and injury-related fatalities in this age group. Approximately one in four older adults will experience a fall annually, highlighting the significance of falls as a public health issue, especially within the aging population.

Approximately 36 million falls are documented among older adults annually, leading to over 32,000 fatalities. Emergency departments treat approximately 3 million older adults each year for fall-related injuries. One out of every five falls results in an injury, such as fractures or head injuries. Hip fractures alone account for at least 300,000 hospitalizations among older individuals annually. Falls are responsible for over 95% of hip fractures, typically occurring from sideways falls. Women experience falls more frequently than men and constitute three-quarters of all hip fractures [14].

1.5 Conclusion

In conclusion, the field of fall detection for elderly individuals and patients has witnessed significant advancements and research efforts due to its substantial demand, potential market, and social value. By understanding the characteristics of falls and employing various detection methods, such as wearable devices, ambiance devices, and camera-based approaches, we can enhance the safety and well-being of the elderly and patients, empowering them to maintain their independence and confidence in daily living.

Chapter 2

State of the art

2.1 Introduction

The fall detection tool proves highly beneficial for elderly individuals, who face an elevated risk of falling due to physical fragility. As people age, falls rise, and extended periods of lying on the ground following a fall can lead to pressure sores, dehydration, and other critical health issues. While there are various fall detection tools accessible, most of them are expensive, stationary, and ineffective. Efficient fall detection systems should be affordable, portable, and accurate, aiming to reduce the impact of delayed medical care and lower the risk of falls.

Approaches based on wearable sensors, wearable sensor-based techniques, and Artificial Intelligence are promising solutions, as non-wearable sensor technology can be intrusive and ineffective in addressing adult isolation [15].

We may distinguish between two major categories of algorithms for wearable technology: threshold-based and machine learning-based algorithms. Although threshold-based algorithms have very good performance in terms of detection accuracy and cheap computing cost, they are particularly challenging to modify for new types of falls and user characteristics. Although machine learning techniques are thought to be more advanced ways to fix this issue, they need a lot of samples to be successful, and there are currently few datasets available to research these events [16].

2.2 Traditional Method (Threshold Method)

2.2.1 Introduction

Numerous methods utilizing portable sensors have been developed recently for the automated detection of falls. Many strategies used the amount of the change in acceleration to estimate falls. Focusing just on high acceleration, however, produces just as many false positives as other activities, including sitting and jogging. Other fall detection methods focus on detecting a change in body position following a fall. These techniques are less successful when the falling posture is not horizontal and may be impacted by activities with comparable posture [17].

When the peak values are either below or above the threshold, the threshold-based fall detection algorithms can distinguish between falls and ADLs. Threshold-based

methods have the benefits of being simple to implement in wearable sensors and having minimal computational complexity. However, because thresholds are created based on the bodily sensations experienced while falling and because set thresholds cannot accommodate all sorts of individual habits of activities in everyday life, threshold-based approaches are not appropriate to detect diverse types of falls [18].

Previous studies demonstrated the use of both an accelerometer and a gyroscope sensor for fall detection. The accelerometer is a useful tool for measuring the impact-induced change in body inertia during falls. In addition, the gyroscope simultaneously offers distinctive knowledge of the body's rotational velocity during a fall occurrence. A fall event causes a significant shift in angular velocity as well as acceleration. Typical everyday activities don't show these changes. Therefore, several accelerations and angular velocity thresholds were established to distinguish between a fall event and an ADL [17].

In a variety of fields of study, including navigation and robotics, as well as in the analysis of human motion, precise orientation measurement is essential. Euler angles, often referred to as Yaw, Pitch, and Roll angles, can be used to depict the spatial orientation to describe the posture of the human body. The present orientation of a body is described by this style of formalism as the result of three fundamental rotations (the angles Yaw, Pitch, and Roll) beginning from a fixed reference frame, which is the world frame. The axes of the fixed reference frame are denoted in Figure 2.1 [19].

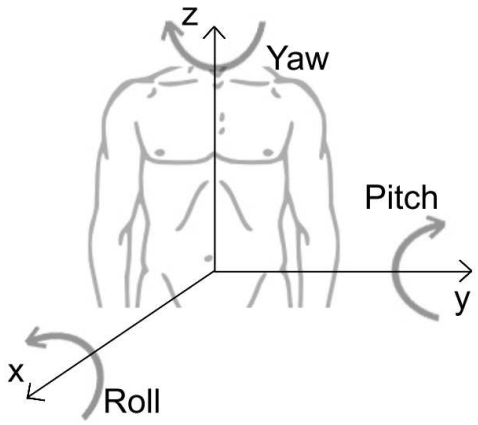

Figure 2.1: Reference system for the Yaw, Pitch, and Roll angles [19].

2.2.2 Sensors Used in Fall Detection Systems

A subject's body orientation cannot be accurately estimated by a single 3-axis accelerometer that is worn around the waist. In reality, the output of a three-axis accelerometer may be used to compute the Pitch and Roll angles to detect a subject's tilt, but it is not feasible to obtain the Yaw angle information. The angle between a fixed heading point (e.g. Earth North) and the device's X-axis is known as yaw.

This is because any rotation of the device around the gravity vector will not affect the accelerometer's output when utilizing gravity as the single reference vector.

An accelerometer alone cannot determine tilt accurately because of inaccuracies brought on by external accelerations and vibrations, which along with gravity makes this instrument unreliable. It is feasible to utilize data from several sensors by employing a suitable algorithm termed an orientation filter to substantially boost the accuracy of the orientation sensing capabilities. In this kind of device, MARG sensors are employed to monitor three-dimensional rotational and translational motions. The orientation filter combines data from the MARG sensor's accelerometer, gyroscope, and magnetometer to produce an accurate assessment of orientation with respect to the Earth's magnetic field and the direction of gravity [19].

Accelerometers

Accelerometers are a form of portable, lightweight inertial sensor that offer a more cost-effective and practical means of evaluation than those often used to create posturographies. An accelerometer consists of a mobile bar hanging by micro-machined springs that oppose the bar's movement (and acceleration). The springs notice their acceleration when this bar deviates. Triaxial accelerometers are used in a device that can support up to three bars with their springs that independently read motions in a dimension and provide activity logs on the three spatial axes [20].

The gravity unit (g), which is based on the acceleration that gravity causes on all objects in ideal conditions (without resistance or friction of any type), serves as the unit of measurement for these devices. Standard gravity or 9.8 meters per second squared (m/s^2), is equated to 1 g of acceleration. The International System of Units uses this as the unit of acceleration measurement.

The use of accelerometers was first restricted to the tracking of physical activity and the assessment of the duration of various activities categorized based on the amount of exercise intensity or resting posture (standing, sitting, or lying down).

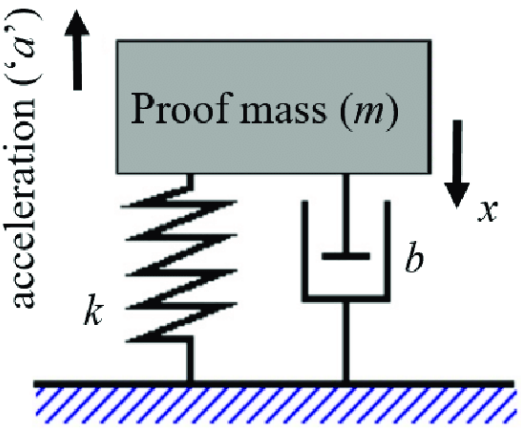

Figure 2.2: Model of accelerometer [21].

Even older people have claimed that these gadgets encourage them to engage in physical exercise. Because accelerometers are portable and may be fastened to the trunk, arms, or any other desirable anatomical place, they have made it possible to measure the motions of a person's everyday life. This feature is extremely valuable in industries like medical, physiotherapy, and sports training since it allows for a tool that can be customized to meet a wide range of measuring goals [20].

This characteristic makes it possible to create a tool that can be tailored to satisfy a variety of measurement objectives, which is particularly beneficial in sectors like healthcare, physiotherapy, and sports training.

Gyroscopes

Gyroscopes are devices affixed to a frame that can detect angular velocity if the frame is rotating. Depending on the underlying physical principle and the underlying technology, there are several classifications of gyroscopes. Gyroscopes can be used independently or as part of more sophisticated systems like the Attitude Heading Reference System, Inertial Measurement Unit, Gyro-compass, and Inertial Navigation System [22].

To measure the angular velocity, MEMS gyroscopes typically utilize a mechanical device that vibrates as a sensor element. Since they don't have spinning components that need bearings, they can be easily miniaturized and made using MEMS device manufacturing methods. The basis for all MEMS gyroscopes with vibrating elements is the transfer of energy between two vibration modes brought on by Coriolis acceleration. An apparent acceleration that may be seen in a rotating frame of reference is the Coriolis acceleration, which is proportional to the angular velocity [22].

To better understand the concept, we can consider a particle of mass (m) moving in space with a velocity (V)

An observer firmly anchored to the z-axis will see the particle moving along the z-axis with a Coriolis acceleration equal to $a_c = 2V * \Omega$ once it is fixed to the reference system in Figure 2.3a and rotating with an angular velocity $\Omega = \Omega_x i$ (with i the unitary vector along the x-axis) around the x-axis, even though no actual force is applied along the z-axis. The vibrating mass MEMS gyroscope's primary physical principle, which may be compared to a mass-spring system, is this Figure 2.3b [22].

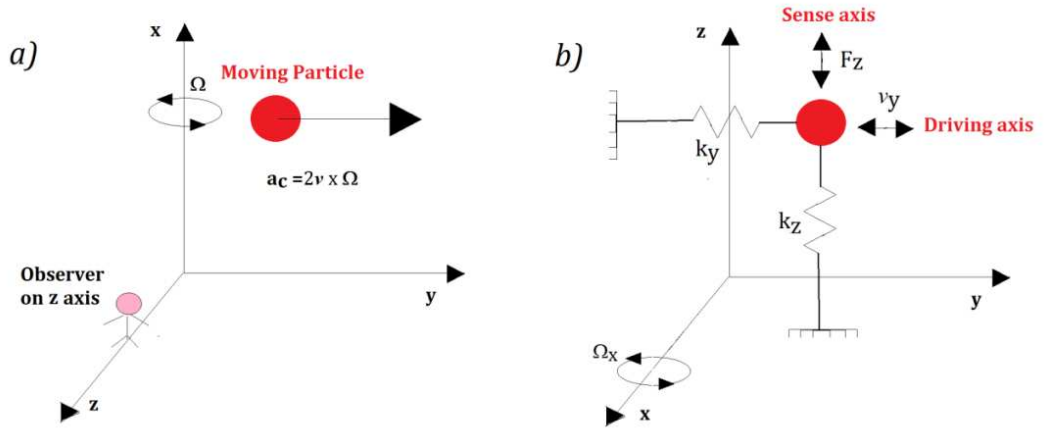

Figure 2.3: (a) Coriolis acceleration (a_c) acting on a moving particle; and (b) mass-spring model of a MEMS gyroscope [22].

Inertial Measurement Units (IMUs)

Contemporary electronics designs have embraced the principles of mechanics and translated them into sensor technology. These sensors can be created using microelectromechanical systems (MEMS). This innovative sensor technology allows for sensor fusion, where multiple sensors and software are combined into a unified package. This approach offers solutions across diverse industries, such as information and communications technology (ICT), the Internet of Things (IoT), and the automotive sector. The integrated solutions are fine-tuned by semiconductor manufacturers, who incorporate embedded compensation and sensor processing alongside a user-friendly programmable interface [23].

MEMS technology enables the integration of precise gyroscopes, accelerometers, magnetometers, and pressure sensors in various axes within a single device. These combined devices, referred to as inertial measurement units (IMUs), are capable of measuring and transmitting information about specific forces, angular velocities, and frequently, the orientation of a body. This technology facilitates the incorporation of multiple sensor types into a unified IMU configuration.

A crucial factor to consider when choosing an IMU is its degree of freedom. IMUs are readily accessible with varying specifications, ranging from two to ten degrees of freedom. The term "freedom" carries diverse meanings based on the context. In this instance, we're not discussing personal or political liberty, but rather referring to the concept in the realm of physics, specifically mechanics. Within mechanics, degrees of freedom pertain to the translational and rotational elements that define the configuration or state of an object. For instance, when dealing with a solid body in space, both translation and rotation consist of three elements each, resulting in a total of six degrees of freedom Figure 2.4.

Combining accelerometers (measuring velocity changes to determine position) and gyroscopes (measuring angular velocity for orientation) enables devices to calculate up to six degrees of freedom. However, the concept of exceeding six degrees of freedom arises from IMU providers recognizing the potential for enhanced performance through additional sensor fusion. By introducing an extra sensor, they enhance accuracy, reduce errors, and gain valuable supplementary data for internal adjustments and compensation. The inclusion of a magnetometer introduces a new layer of sensor data by detecting Earth's magnetic field, enabling the determination of directional heading. When this magnetometer data is integrated with accelerometer and gyroscope data, manufacturers claim an additional three degrees of freedom. This gives rise to the creation of a nine-degree of freedom IMU [23].

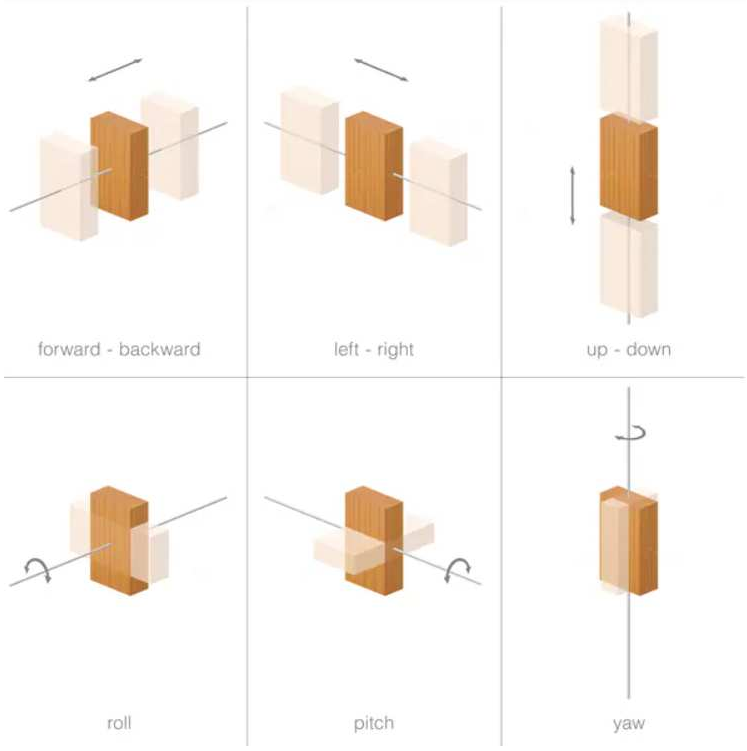

Figure 2.4: Six degrees of freedom. Possibilities of movement of a rigid body in 3D space. Forward, backward, left, right, up, and down, plus rotations about the three axes [23].

2.3 Machine Learning

2.3.1 Introduction

In recent years, the field of machine learning (ML) has undergone a remarkable evolution, revolutionizing data analysis and intelligent computing applications [24]. As data availability continues to surge and computational capabilities advance, machine learning has emerged as a pivotal technology capable of addressing complex

challenges across diverse domains.

Unlike traditional programming paradigms, where explicit instructions are provided to accomplish tasks, machine learning empowers systems to learn from experience and enhance performance iteratively [25]. This shift has led to a surge in interest and adoption of ML techniques, driven by their ability to tackle intricate problems that traditional methods struggle to handle.

Machine learning encompasses various learning algorithms, broadly categorized into supervised, unsupervised, semi-supervised, and reinforcement learning [26]. These categories cater to a wide array of scenarios, from training models with labeled data to discovering hidden patterns and enabling systems to make sequential decisions.

As we delve deeper into this field, we'll explore the nuances of these learning categories and delve into their applications in real-world scenarios. These learning methods are becoming more and more popular every day which is shown in Figure 2.5 [25].

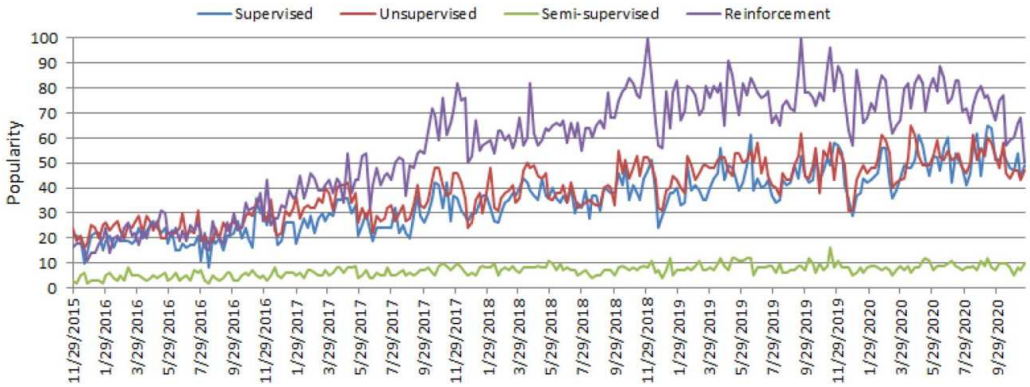

Figure 2.5: The worldwide popularity score of various types of ML algorithms (supervised, unsupervised, semi-supervised, and reinforcement) in a range of 0 (min) to 100 (max) over time where the x-axis represents the timestamp information and the y-axis represents the corresponding score [25].

2.3.2 Types of Real-World Data and Machine Learning Techniques

Algorithms for machine learning often ingest and analyze data to discover patterns relating to people, business processes, transactions, events, and so on. Following, we go through major kinds of machine learning methods and types of real-world data [25].

Types of Real-World Data

In the realm of machine learning models and data-driven real-world systems, the indispensability of data is widely acknowledged. This data can take various forms, including structured, semi-structured, and unstructured data. Additionally, a distinct category known as 'metadata' plays a crucial role. Metadata encompasses information

that provides insights into the data itself, offering contextual details, attributes, and characteristics. It often includes specifics such as the data collection timeframe, the entity responsible for the collection, the semantic interpretation of various dataset fields, and the overall data structure. By furnishing this contextual layer, metadata aids users in comprehending, interpreting, and efficiently managing the primary dataset [25].

Structured: It is utilized by an entity or computer program and has a clearly defined structure, complies with a data model that follows a standard order, is very well-organized, and is simple to access. Structured data is often kept in a tabular manner in well-defined systems like relational databases. Structured data includes things like names, dates, addresses, credit card numbers, stock information, geolocation, etc.

Unstructured: Conversely, unstructured data lacks a predetermined structure or arrangement, posing greater challenges in terms of collection, processing, and analysis. It predominantly comprises textual and multimedia content, rendering its handling and interpretation more complex. Instances of unstructured data encompass a wide array of sources, including sensor data, emails, blog posts, wikis, word processing documents, PDFs, audio and video files, images, presentations, web pages, and a multitude of other business-related documents.

Semi-structured: Semi-structured data does not reside in a relational database like the previously described structured data, but it does possess some organizational characteristics that facilitate analysis. Semi-structured data includes but is not limited to, documents in HTML, XML, JSON, and NoSQL databases.

Metadata: It is "data about data," not the typical form of data. The main distinction between "data" and "metadata" is that data are only the elements that may categorize, quantify, or even document anything in relation to the data attributes of an organization. However, metadata provides greater context for data consumers by describing the pertinent data facts. The author, file size, date the document was created, keywords used to characterize the content, etc. are some simple examples of a document's metadata.

Types of Machine Learning Techniques

Supervised learning, Unsupervised learning, Semi-supervised learning, and Reinforcement learning are the four main classes of machine learning algorithms as shown in Figure 2.6. Following this, we quickly go through each sort of learning method and the extent to which it may be used to address difficulties in the actual world [25].

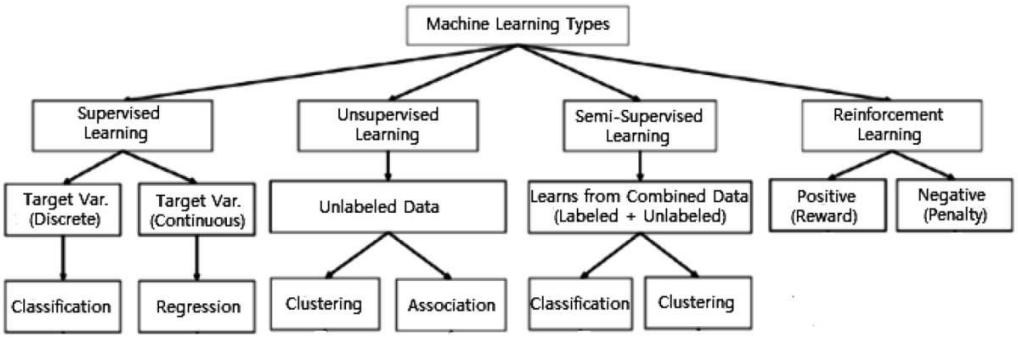

Figure 2.6: Various types of machine learning techniques [25].

Supervised: Supervised learning involves the machine learning process of acquiring knowledge from sample input-output pairs to establish a functional relationship between inputs and outputs. This is achieved by utilizing labeled training data and a pool of training instances to deduce a predictive function. Supervised learning is employed when specific objectives are defined to be achieved from a given set of inputs, reflecting a goal-oriented strategy. Common tasks within supervised learning encompass "classification," which segregates data into categories, and "regression," which models data trends. To illustrate, an instance of supervised learning involves foreseeing the class label or sentiment associated with the text, such as a tweet or a product review—referred to as text classification.

Unsupervised: Unsupervised learning involves the examination of datasets lacking predefined labels, and it operates without human intervention, representing a process driven solely by the data itself. This approach is extensively applied for extracting inherent patterns, uncovering significant trends and structures, as well as discovering natural groupings within outcomes, all within an exploratory context. Predominant unsupervised learning tasks encompass clustering, density estimation, feature learning, dimensionality reduction, identification of association rules, and anomaly detection.

Semi-supervised: Semi-supervised learning can be described as an amalgamation of the aforementioned supervised and unsupervised approaches, functioning with both labeled and unlabeled datasets. As such, it occupies an intermediate position between learning "without supervision" and learning "with supervision." In practical scenarios, instances of labeled data can be scarce in various contexts, while unlabeled data are abundant, rendering semi-supervised learning valuable. The overarching objective of a semi-supervised learning model is to yield superior predictive outcomes compared to those attainable solely through the use of labeled data in the model. Several application domains make use of semi-supervised learning, including machine translation, fraud detection, data labeling, and text classification.

Reinforcement: Reinforcement learning stands as a distinct category within the realm of machine learning algorithms, empowering software agents and machines to autonomously assess optimal actions within specific contexts or environments, all to enhance their effectiveness—an approach rooted in the dynamics of the environment itself. This learning paradigm hinges upon the concepts of rewards and penalties, with its ultimate objective being to leverage insights garnered from interactions with the environment to make decisions that maximize rewards or mitigate risks. This methodology serves as a potent tool for training AI models that contribute to increased automation or heightened operational efficiency within intricate systems such as robotics, autonomous driving, manufacturing, and supply chain logistics. However, it's important to note that its application might not be the most suitable choice for addressing fundamental or straightforward problems.

As a result, depending on the nature of the data stated earlier and the desired result, various machine-learning approaches can play a key role in the development of effective models in a variety of application areas. Table 2.1 provides an overview of several machine-learning approaches with illustrations.

Learning type	**Model building**	**Examples**
Supervised	Algorithms or models learn from labeled data (task-driven approach)	Classification, regression
Unsupervised	Algorithms or models learn from unlabeled data (Data-Driven Approach)	Clustering, associations, dimensionality reduction
Semi-supervised	Models are built using combined data (labeled + unlabeled)	Classification, clustering
Reinforcement	Models are based on reward or penalty (environment-driven approach)	Classification, control

Table 2.1: Various types of machine learning techniques with examples [25].

2.3.3 Learning algorithms

An algorithm that can learn from data is called a machine learning algorithm. How do we define learning, though? If a computer program's performance at tasks in a class of tasks T, as measured by P, gets better with experience E, it is said to learn from experience E with regard to those tasks T and performance measure P. The parts that follow offer clear explanations and illustrations of the many tasks, performance measures, and experiences that may be utilized to create machine learning algorithms [27].

The Task, T

Machine learning allows us to tackle tasks that are too difficult to solve with fixed programs written and designed by human beings. From a scientific and philosophical point of view, machine learning is interesting because developing our understanding of machine learning entails developing our understanding of the principles that underlie intelligence.

In this relatively formal definition of the word "task," the process of learning itself is not the task. Learning is our means of attaining the ability to perform the task. For example, if we want a robot to be able to walk, then walking is the task. We could program the robot to learn to walk, or we could attempt to directly write a program that specifies how to walk manually.

Machine learning tasks are usually described in terms of how the machine learning system should process an example. An example is a collection of features that have been quantitatively measured from some object or event that we want the machine learning system to process. We typically represent an example as a vector $x \in \mathbb{R}^n$ where each entry x_i of the vector is another feature. For example, the features of an image are usually the values of the pixels in the image [27].

Many kinds of tasks can be solved with machine learning. Some of the most common machine-learning tasks include the following:

- Classification
- Classification with missing inputs
- Regression
- Transcription
- Machine translation
- Structured output
- Anomaly detection
- Synthesis and sampling
- Imputation of missing values
- Denoising
- Density estimation or probability mass function estimation

Given that this project revolves around a classification task, we will focus exclusively on exploring and addressing this particular challenge.

Calssification In this form of task, the computer program is tasked with determining the specific category, out of k possible categories, to which a given input belongs. To tackle this task, the learning algorithm is commonly required to generate a function $f : \mathbb{R}^n \to \{1, \ldots, k\}$. When $y = f(x)$, the model assigns an input characterized by the vector x to a category denoted by the numerical code y. Variations of the classification task exist; for instance, where f produces a probability distribution spanning the classes.

An illustrative instance of a classification task is object recognition. Here, the input comprises an image, often represented as an array of pixel brightness values, and the output entails a numeric code that identifies the object present in the image [27].

The Performance Measure, P

To evaluate the abilities of a machine learning algorithm, we must design a quantitative measure of its performance. Usually, this performance measure P is specific to the task T being carried out by the system.

For tasks such as classification, classification with missing inputs, and transcription, we often measure the accuracy of the model. Accuracy is just the proportion of examples for which the model produces the correct output. We can also obtain equivalent information by measuring the error rate, and the proportion of examples for which the model produces incorrect output. We often refer to the error rate as the expected 0-1 loss. The 0-1 loss on a particular example is 0 if it is correctly classified and 1 if it is not. For tasks such as density estimation, it does not make sense to measure accuracy, error rate, or any other kind of 0-1 loss. Instead, we must use a different performance metric that gives the model a continuous-valued score for each example. The most common approach is to report the average log probability the model assigns to some examples.

Usually, we are interested in how well the machine learning algorithm performs on data that it has not seen before since this determines how well it will work when deployed in the real world. We, therefore, evaluate these performance measures using a test set of data that is separate from the data used for training the machine learning system.

The choice of performance measure may seem straightforward and objective, but it is often difficult to choose a performance measure that corresponds well to the desired behavior of the system.

In some cases, this is because it is difficult to decide what should be measured. For example, when performing a transcription task, should we measure the accuracy of the system at transcribing entire sequences, or should we use a more fine-grained performance measure that gives partial credit for getting some elements of the sequence correct? When performing a regression task, should we penalize the system more if it frequently makes medium-sized mistakes or if it rarely makes very large

mistakes? These kinds of design choices depend on the application.

In other cases, we know what quantity we would ideally like to measure, but measuring it is impractical. For example, this arises frequently in the context of density estimation. Many of the best probabilistic models represent probability distributions only implicitly. Computing the actual probability value assigned to a specific point in space in many such models is intractable. In these cases, one must design an alternative criterion that still corresponds to the design objectives, or design a good approximation to the desired criterion [27].

2.3.4 Capacity, Overfitting, and Underfitting

The main difficulty in machine learning is that we must be able to recognize new inputs that have never been seen before, in addition to those on which our model was trained. Generalization is the capacity to perform effectively on previously unseen inputs.

Usually, to train a machine learning model, we have access to a training set, which allows us to compute an error measure called the training error on the training set and then minimize it. What we have so far discussed is merely an optimization issue. The fact that we also want the generalization error, sometimes known as the test error, to be minimal distinguishes machine learning from optimization. The expected value of the error on a new input is what is referred to as the generalization error.

In this case, the expectation is spread across a variety of potential inputs, chosen from the range of inputs we anticipate the system would experience in the real world. Typically, we evaluate a machine learning model's performance on a test set of examples that were gathered separately from the training set to determine the generalization error of the model.

Of course, we do not preset the parameters beforehand and sample both datasets when using a machine-learning technique. To decrease training set error, we sample the training set, utilize that information to determine the parameters, and then sample the test set. The expected test error in this method is more than or equal to the expected value of the training error. A machine learning algorithm's performance is influenced by its capacity to

1. Make the training error small.
2. Make the gap between training and test error small.

Underfitting and overfitting, the two main problems with machine learning, are represented by these two variables. When the model is unable to achieve a suitably low error value on the training set, underfitting occurs. When the difference between the training error and test error is too great, overfitting occurs. By changing a model's capacity, we may influence whether it is more likely to overfit or underfit. A model's capacity is officially defined as its ability to suit a wide range of functions. Low-capacity models could have trouble fitting the training set. Models with large capacity may overfit by remembering training set characteristics that do not help them on the test set [27].

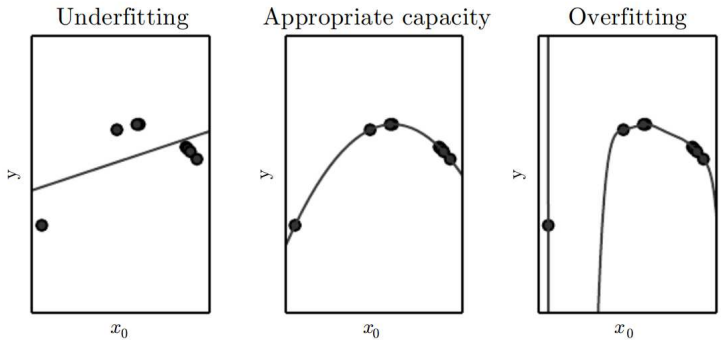

Figure 2.7: Figure showing: Underfitting, overfitting, and the optimal capacity [27].

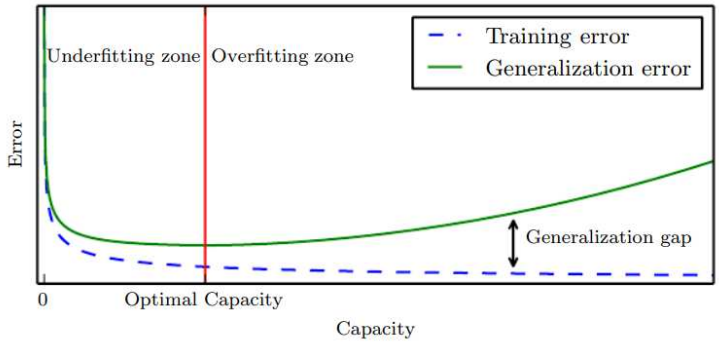

Figure 2.8: Typical relationship between capacity and error. Training and test errors behave differently. At the left end of the graph, training error and generalization error are both high. This is the underfitting regime. As we increase capacity, training error decreases, but the gap between training and generalization error increases. Eventually, the size of this gap outweighs the decrease in training error, and we enter the overfitting regime, where the capacity is too large, above the optimal capacity [27].

2.3.5 Hyperparameters

The majority of machine learning algorithms contain several options that we may use to regulate the algorithm's behavior. They are referred to as hyperparameters. The learning algorithm occasionally selects a setting as a hyperparameter that it does not learn because it is hard to optimize. Because it is inappropriate to learn that hyperparameter on the training set, the setting is typically a hyperparameter. All hyperparameters that affect model capacity fall under this. Such hyperparameters would always select the largest model capacity if they were learned on the training set, leading to overfitting to solve this problem we use a Validation set [27].

2.3.6 Validation Sets

The test examples must remain untouched when making decisions about the model, including its hyperparameters. As a result, no instance from the test set should be integrated into the validation set. As a remedy, the validation set is exclusively formed from the training data. The process involves splitting the training data into two distinct subsets. One subset facilitates parameter learning, while the other serves as the validation set, aiding in estimating the generalization error during or post-training and guiding updates to hyperparameters.

Although the data subset employed for parameter learning is still commonly referred to as the training set, this term may lead to confusion with the broader dataset employed throughout the entire training procedure. The subset responsible for shaping hyperparameter choices is termed the validation set. Typically, around 80% of the training data is dedicated to actual training, with the remaining 20% allocated to validation. Since the validation set plays a role in "training" the hyperparameters, the validation set's error will likely underestimate the generalization error, although typically to a lesser extent than the training error. Once hyperparameter optimization concludes, the generalization error can be approximated using the test set [27].

2.4 Deep learning

Deep learning enables computational models to learn progressively abstract representations of data by utilizing multiple layers of processing. These techniques have led to significant advancements in various fields, including speech recognition, visual object identification, object detection, and even specialized areas like drug discovery and genomics. By employing the backpropagation algorithm, deep learning algorithms can decipher intricate patterns within extensive datasets. This algorithm guides the adjustments of internal parameters in each layer, helping the machine transform representations from the previous layer into those in the current layer. Notably, deep convolutional networks have revolutionized the analysis of images, videos, speech, and audio. Additionally, recurrent networks have shed light on understanding sequential data like text and speech [28].

2.4.1 Artificial neural networks

A neural network refers to a data processing setup comprising numerous basic interconnected processing units, designed based on the brain's cerebral cortex structure. As a result, neural networks can excel in tasks that humans or animals perform adeptly, areas where traditional computers often struggle. Recently, neural networks have gained significant attention as a promising field for research, advancement, and solving various practical challenges [29].

The basic cellular building block of the neurological system in the brain is known as a neuron. It is a straightforward processing unit (soma) that collects and integrates

data from neighboring neurons through input dendrites, which include synaptic connections. Figure 2.9 depicts the fundamental elements of a neuron, and Figure 2.10 shows their schematic counterparts. The neuron "fires" and sends an output signal up the axon if the sum of all the dendritic signals is strong enough. Through synapses, which are connections carrying a neurotransmitter fluid that regulates the flow of electrical signals, the axon divides and joins hundreds of dendrites (input routes) of other neurons. The synaptic strength of the synaptic connections determines the amplitude of the impulses that are sent across the synapses, which are electrochemical in nature. As the brain "learns," the strength or conductance of a synaptic connection changes (conductance is the inverse of resistance). In other words, the brain's basic "memory units" are its synapses [29].

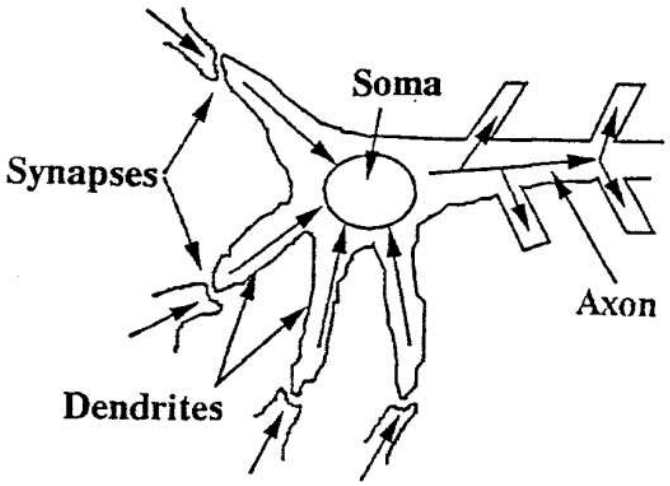

Figure 2.9: Sketch of a Neuron Showing Components. [29].

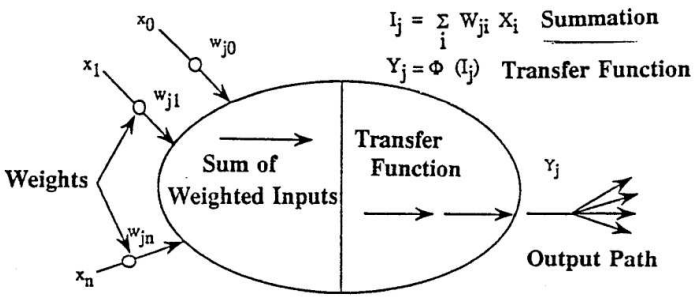

Figure 2.10: Schematic Representation of an Artificial Neuron [29].

The artificial neural systems used to simulate this brain function on computers typically consist of several artificial neurons, also known as processing elements or neurodes. These processing units resemble neurons in that they both aggregate (sum

up) the input data and contain several inputs (dendrites). After that, this sum is placed through a nonlinear filter known as a transfer function, which is often a bias or a threshold function in which output signals are only produced if the output exceeds the threshold value. The output can also be a continuous function of the total input, commonly a sigmoid function restricted to the range 0 to +1 or an arctangent or hyperbolic tangent function restricted to the range -1 to +1 [29].

2.4.2 Activation functions

There are many choices of the activation function used by artificial neural networks:

Sigmoid function

A non-linear activation function called the Sigmoid is frequently employed in feed-forward neural networks. It is a bounded differentiable real function with positive derivatives everywhere and a certain amount of smoothness, defined for real input values. The Sigmoid function is given by the relationship Eq.(2.1):

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.1}$$

This type of activation function is primarily utilized in shallow networks. One of its key advantages lies in its simplicity and ease of comprehension. It has been effectively applied in binary classification problems, demonstrating successful outcomes [30].

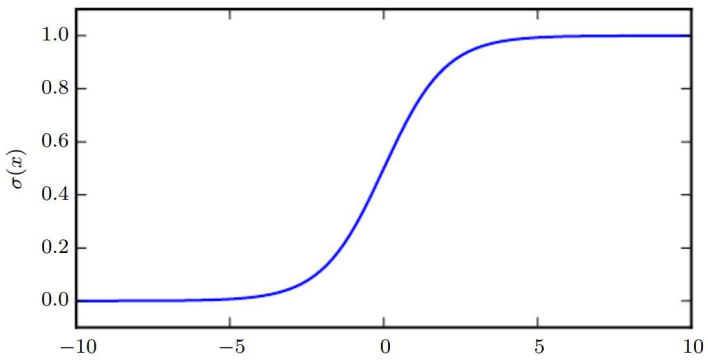

Figure 2.11: The logistic sigmoid function [27].

The Sigmoid activation function has significant drawbacks, including gradient saturation, slow convergence, non-zero centered output, sharp damp gradients during backpropagation from deeper hidden layers to the input layers, and non-zero centered output, which causes the gradient updates to propagate in different directions. To address some of these issues with the Sigmoid activation function, other types of activation functions, such as the hyperbolic tangent function, were proposed [30].

Hyperbolic Tangent Function (Tanh)

This type of activation function is smoother, zero centered and it ranges between -1 to 1, and it's given by the Eq.(2.2)

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.2}$$

Due to its superior training performance for multi-layer neural networks, the tanh function has replaced the sigmoid function as the favored function. However, the vanishing gradient issue that the sigmoid functions also faced could not be resolved by the tanh function. The function's primary benefit is that it generates zero-centered output, which helps the back-propagation process. The tanh function has the peculiarity that it can only achieve a gradient of 1 when the input value is 0, or when x is zero. As a result, the computation of the tanh function results in some dead neurons. The rectified linear unit (ReLU) activation function was created as a result of more research into activation functions to address the tanh function's drawback [30].

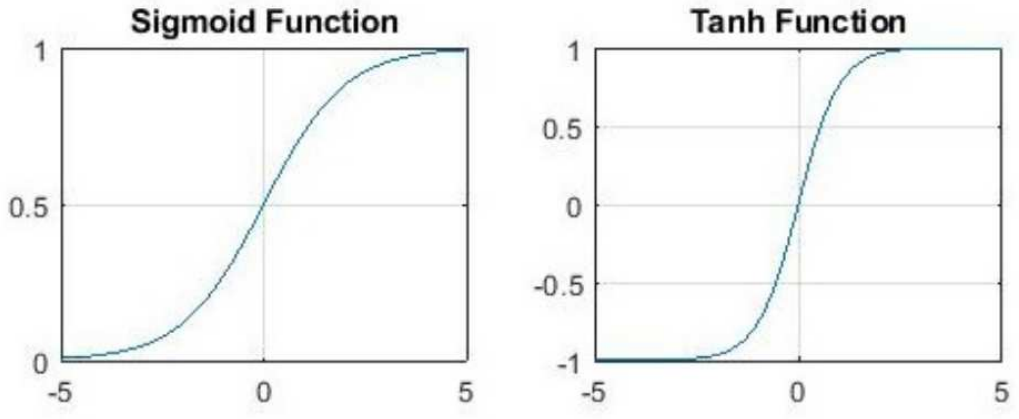

Figure 2.12: Pictorial representation of Sigmoid and Tanh activation function responses [30].

Softmax Function

Another form of activation function utilized in neural computing is the Softmax function. A vector of real values is utilized to compute the probability distribution. The output of the Softmax function is a range of values between 0 and 1, with the probability total being equal to 1. and it's given by the Eq.(2.3).

$$\mathrm{f}(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}} \tag{2.3}$$

In multi-class models, the Softmax function is used to return probabilities for each class, with the target class having the highest probability. The Sigmoid and Softmax

activation functions vary primarily in that the former is used for binary classification jobs while the latter is utilized for multivariate classification tasks [30].

Rectified Linear Unit (ReLU) Function

Since it was first suggested and has continued to this day, the rectified linear unit (ReLU) activation function has been the most popular activation function for deep learning applications with the most cutting-edge outcomes. A quicker learning activation function is the ReLU. When compared to Sigmoid and Tanh activation functions, it delivers greater deep learning performance and generalization. Since the ReLU represents a nearly linear function, it retains the characteristics of linear models that made them simple to optimize using gradient-descent techniques [30].

Each input element is subjected to a threshold operation by the ReLU activation function, which sets values less than zero to zero. As a result, the ReLU is given by Eq.(2.4)

$$f(x) = \max(0, x) = \begin{cases} x_i, & \text{if } x_i \geq 0 \\ 0, & \text{if } x_i < 0 \end{cases} \tag{2.4}$$

This function eliminates the vanishing gradient issue seen in earlier forms of activation function by rectifying the values of the inputs less than zero and driving them to zero. The primary benefit of rectified linear units in computation is that they ensure quicker calculation because they do not compute exponentials or divisions, which increases computation speed overall [30].

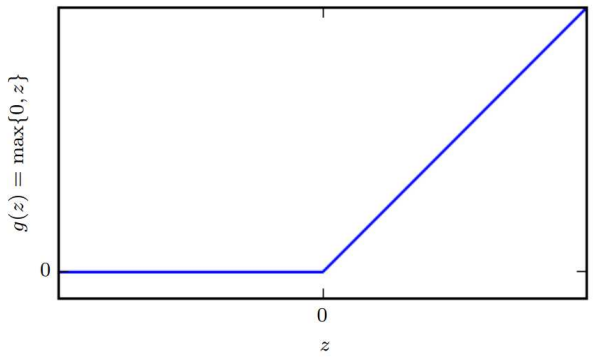

Figure 2.13: The rectified linear activation function [27].

2.4.3 Feedforward networks

Deep feedforward networks, commonly known as feedforward neural networks or multilayer perceptrons (MLPs), serve as fundamental models in the realm of deep learning. The primary objective of a feedforward network is to create an approximation of a given function f^*. In the context of classification, where an input x is associated with a category y, the network establishes a mapping denoted as $y = f(x; \theta)$, where θ represents the parameters. Through learning, the network seeks

to determine the optimal values of these parameters θ that lead to the most accurate approximation of the desired function [27].

The reason these models are named feedforward is that data moves from the input x via the function being evaluated, the calculations necessary to determine f, and ultimately to the output y. The model's outputs cannot be fed back into it since there are no feedback links [27]. in Figure 2.14 an example of a shallow and a deep architecture is shown [31].

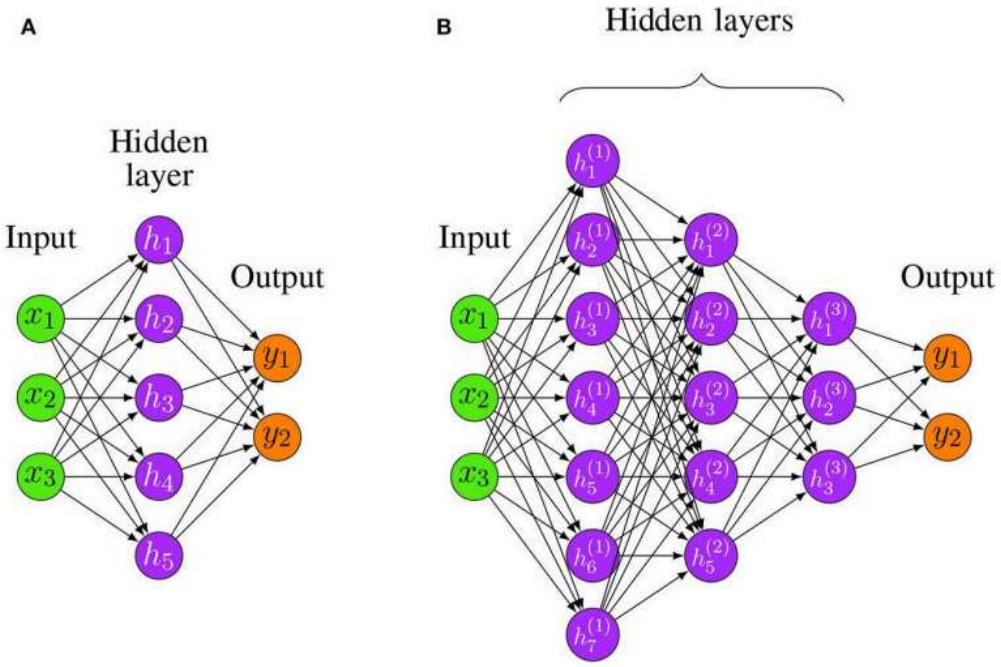

Figure 2.14: Two examples for Feedforward Neural Networks. (A) A shallow FFNN. (B) A Deep Feedforward Neural Network (D-FFNN) with 3 hidden layers [31].

In general, a network's depth refers to the amount of nonlinear transformations that occur between its separating layers, whereas a hidden layer's width refers to the dimensionality of its hidden neurons. In contrast to the shallow design in Figure 3A, which has a depth of 2, Figure 3B has a depth of 4 (four layers overall, minus the input layer). Although the minimum requirement for a Feedforward Neural Network (FFNN) design to be called deep is controversial, systems with more than two hidden layers are typically regarded as such [31].

The activation functions of a feed-forward neural network, commonly known as an MLP or multilayer perceptron, can be linear or non-linear [27]. It is important that the NN does not contain any cycles that would permit direct feedback. Eq.(2.5) explains how an MLP generates its output from its input.

$$f(x) = \phi^{(2)}(W^{(2)}\phi^{(1)}(W^{(1)}x + b^{(1)}) + b^{(2)}) \tag{2.5}$$

A learning rule is necessary for determining the ideal parameters. Defining an error function (or cost function) and an optimization algorithm together is a typical strategy for determining the best parameters by reducing the error for training data [31].

2.4.4 Dropout layer

Dropout functions as a regularization technique for fully connected layers within neural networks. In this method, every element of a layer's output is retained with a probability p, while with a probability of $(1 - p)$, it is set to 0. Empirical studies demonstrate that employing dropout enhances the network's capability to generalize, leading to improved performance during testing [32].

2.4.5 Dense layer

The fundamental units of neural networks are dense layers. They are made up of a group of neurons, each of which is connected to every neuron in the layer below. The word "dense" describes how each neuron is closely coupled to every other neuron in the layer below [33].

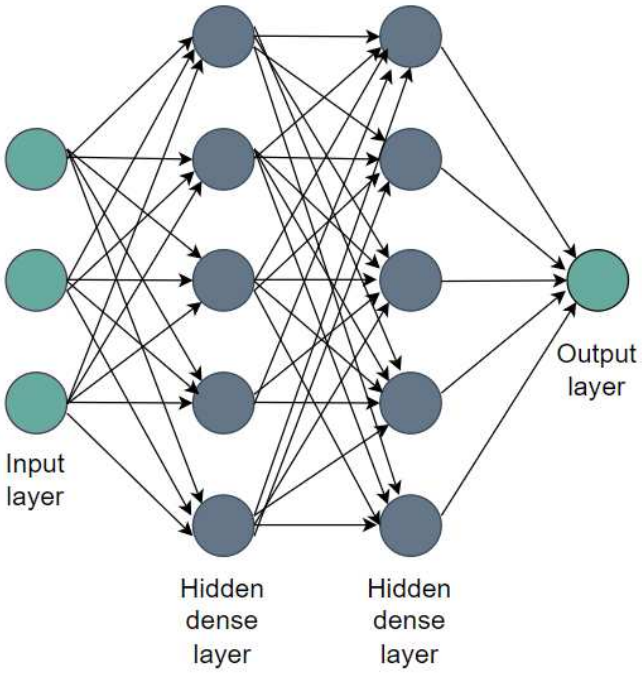

Figure 2.15: Layers in a neural network [33].

Dense layers play a crucial role in neural networks by capturing complex patterns and relationships in data. With the ability to configure the number of units, activation

functions, and other parameters, dense layers provide flexibility and power in building deep learning models for various tasks [33].

2.4.6 Convolutional Networks

Convolutional networks, commonly referred to as CNNs or convolutional neural networks, are a particular class of neural networks used to process input with a predetermined, grid-like architecture. Time-series data, which may be seen as a 1D grid capturing samples at predetermined intervals, and picture data, which can be shown as a 2D grid of pixels, are two examples. Convolutional networks have achieved great success in real-world settings. The phrase "convolutional neural network" refers to a network that uses the convolution mathematical technique. A specific kind of linear processing is convolution. Convolutional networks are simple neural networks that, in at least one of their layers, employ convolution rather than standard matrix multiplication [27].

Convolution: a set of convolutional filters is applied to the input images, each of which activates a different feature of the images.

Pooling: reduces the number of parameters the network needs to learn about by performing nonlinear downsampling on the output [34].

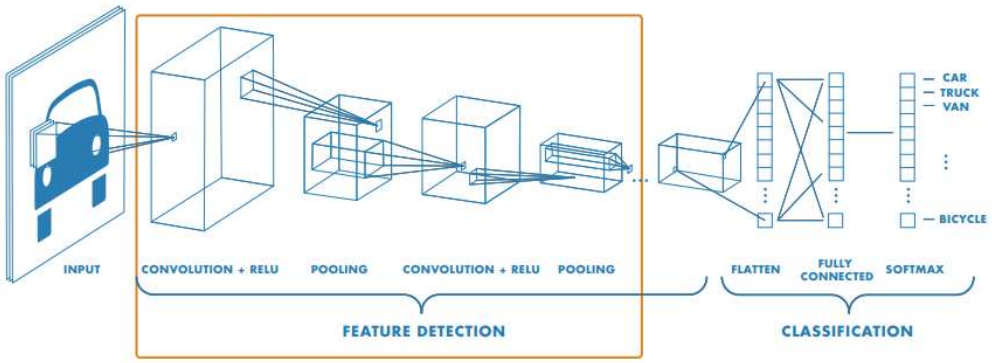

Figure 2.16: Layers in a CNN [34].

2.5 Conclusion

In this chapter, we have delved into the current state of the art in fall detection. We began by discussing the threshold method, which relies on a predefined set of thresholds to identify falls. Subsequently, we transitioned to the realm of machine learning, where algorithms learn to discern fall patterns from the input signals. Lastly, we explored more intricate approaches involving deep learning, particularly the utilization of feedforward neural networks. In the upcoming chapter, we will

present a comprehensive review of fall detection methodologies encompassing both machine learning and deep learning techniques.

Chapter 3

Literature review

3.1 Introduction

Over the past two decades, extensive research has been conducted on fall detection and fall prevention strategies, which are crucial for addressing the problem of falls among the elderly. Researchers have thoroughly investigated fall detection methods, exploring various approaches. These methods involve the utilization of diverse sensors to gather valuable signals for subsequent processing and analysis. Additionally, different analysis algorithms are employed to handle the collected data effectively. The majority of fall detection systems typically utilize accelerations caused by the impact on the body for shock detection [35].

Generally, fall detection systems rely on accelerations to detect the shock resulting from body impact. Article [36] considered only the accelerometer sensor embedded in the smartphone to detect *Falls* and *ADLs*. To enhance the performance of fall detectors, researchers have explored various strategies. Some have incorporated additional sensors, while others have employed sophisticated machine learning-based processing techniques. These approaches aim to improve the accuracy and reliability of fall detection systems. For example, [37, 38, 39, 40, 41, 42] combine a tri-axial accelerometer and gyroscope to detect falls. While some articles for example [43] and [44] use machine learning to effectively detect falls.

In addition to fall detection systems, numerous other fall-related technologies have been developed. These include fall prevention methods, the use of low-power technologies for fall detectors, and the identification of optimal sensor locations to achieve high accuracy in fall detection. These advancements aim to address different aspects of fall prevention and detection, contributing to the overall improvement of safety for individuals at risk of falling.

3.2 Method

The *Mdpi*, *Pubmed*, *IEEExplore*, *Google scholar*, *ResearchGate* search engines were used to carry out the search for the selected research, the keywords for this task were: "Fall detection", "elderly fall detection", "embedded", "real-time", "wearable", "algorithm", "microcontroller", "embedded". we excluded research that focuses on

video and audio fall detection solutions and included only the research that uses embedded sensors such as an accelerometer, gyroscope, and magnetometer). Only studies from 2017 to 2023 were included in the review.

3.3 Results

The aforementioned search method resulted in the identification of ten research articles published between 2017 and 2023. These articles are relevant to the topic at hand and contribute to the current body of knowledge in the field. By employing this approach, recent and up-to-date research findings were included in the review.

3.3.1 Marques J et al. (2023)

This study [45] introduces a novel wrist-based dataset aimed at addressing the fall detection problem. The dataset is utilized to conduct extensive research utilizing the FS-1 feature set, comprising minimal computational features such as maximum, minimum, mean, and variance. Various machine-learning techniques are explored within the constraints of battery and memory limitations.

The core of this research revolves around an accelerometer-based fall detection system integrated into a smartwatch, sampling data at 40 Hz. The system employs the FS-1 feature set for training a 3-nearest neighbor (3NN) algorithm using Euclidean distance. A window size of 9 seconds is chosen, considering practical battery and memory constraints. An innovative learning version of the algorithm is also developed, progressively enhancing fall detection performance over time. Impressively, this approach achieves zero instances of false positives or false negatives across a span of four days.

In the context of modern smartwatches, positioned for comfort on the wrist, this study seeks to establish an efficient fall detection mechanism. Leveraging a 3D accelerometer, gyroscope, and orientation sensors but using only an accelerometer for training to achieve higher accuracy and lower power consumption during testing, the study formulates fall detection as a binary classification task. The primary objectives encompass achieving real-time fall detection while minimizing false positive outcomes. Furthermore, recognizing the scarcity of wrist-based fall detection data, especially about the elderly, the study introduces a new dataset encompassing daily activities commonly performed by the elderly population.

The selection of fall types is supported by a survey that recognizes contextual and causal factors contributing to falls. This work conscientiously preserves the distinct directions of falls (forward, backward, and lateral) and embraces prevalent natural scenarios. The chosen falls are categorized based on their causes such as "Fall forward while walking caused by a slip" and "Fall forward while sitting, caused by fainting or falling asleep".

Considering activities of daily living (ADLs), the study builds upon the SisFall

dataset. The ADLs are selected based on their real-life frequency and resemblance to fall-like movements that could lead to false positives. These selections align with the wrist-based focus of this project, taking into account the distinct movements and stationary position of the wrist. the chosen ADLs, encompassing various activities such as walking, jogging, stair negotiation, sitting, crouching, and more.

Various experimentation iterations involving distinct window sizes, sensor combinations, frequencies, and machine learning algorithms have been undertaken in pursuit of identifying the optimal configuration for maximizing accuracy. Remarkably, the culmination of these trials has pinpointed a specific configuration as the most accurate. This configuration entails employing a window size of 9 seconds, utilizing solely accelerometer data, applying the 3-nearest neighbor (3NN) algorithm, and extracting essential features comprising Maximum, Minimum, Mean, and Variance. Additionally, a frequency of 40 Hz has been identified as integral to achieving this peak accuracy level. This meticulous exploration underscores the meticulous approach undertaken to optimize the fall detection system's performance.

In summary, this study's contributions encompass a novel wrist-based dataset, an innovative fall detection approach integrating low computational features, and a comprehensive analysis of machine learning methodologies. The research tackles fall detection within the context of wearable smartwatches, aiming for real-time accuracy while accommodating practical limitations. Furthermore, the study enriches the field by introducing wrist-based datasets, thus addressing a prevailing data scarcity in fall detection research.

3.3.2 Lee Y et al. (2023)

This article [43] presents a comprehensive overview of a comfortable, cost-effective, and reliable fall detection system for older adults. The research addresses the lack of automatic fall detection systems specifically designed for this population. The study introduces a wireless, flexible, skin-wearable electronic device capable of accurate motion sensing while ensuring user comfort. The device utilizes thin copper films and incorporates a six-axis motion sensor. It is directly laminated on the skin without the need for adhesives, enabling the collection of precise motion data.

To evaluate fall detection accuracy using the proposed device, the researchers explore various deep learning models, body locations for device placement, and input datasets based on different human activities. Results indicate that placing the device on the chest achieves optimal accuracy of over 98% for fall detection using motion data from older adults. The study emphasizes the importance of collecting large motion datasets directly from older adults to enhance the accuracy of fall detection for this population.

The skin-wearable motion monitoring device shown in Figure 3.2 is designed to be ultrathin, flexible, and comfortable. User perception surveys conducted with older adults participating in the study confirm the device's successful integration with the

skin without the use of adhesives. The antenna design of the device is also optimized to ensure seamless wireless communication even during various human activities.

For fall classification using deep learning models, the study explores different models, input datasets, and body locations for device placement. The LSTM model trained on XYZ accelerometer and gyroscope input datasets shown in Figure 3.1 demonstrates the highest accuracy of 97.6% and 98.5% for fall detection in young and older adults, respectively. The chest is identified as the optimal location for placing the skin-wearable motion monitoring device.

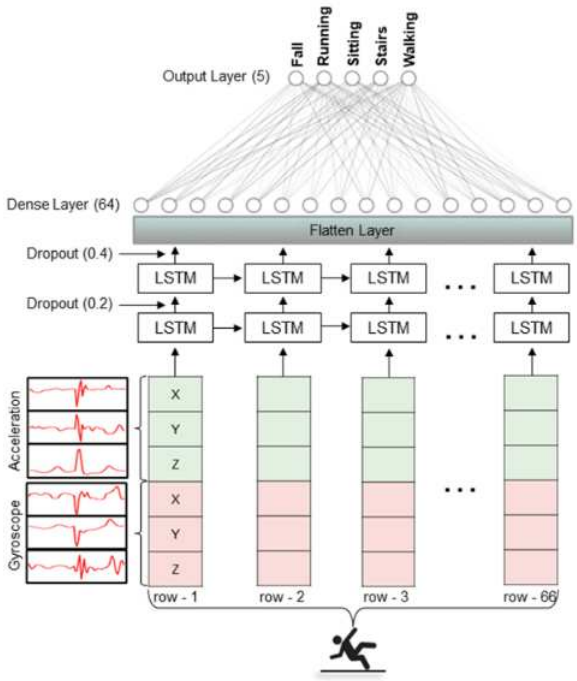

Figure 3.1: Overall architecture of the LSTM-based classification model used by Lee Y et al. (2023). The input was fed to two LSTM layers followed by dropouts for regularization purposes. Finally, outputs from all the LSTM units were merged in a one-dimensional vector using a flatten layer which was followed by a hidden layer with 64 neurons [43].

The article highlights the significance of actively involving older adults in building large motion databases to improve the sensitivity and specificity of deep learning models for reliable fall and activity classifications. Encouraging active participation can be achieved through fall risk education, raising awareness about the seriousness of falls as public health issues, and emphasizing how wearable technologies can mitigate the adverse consequences of falls for older adults.

Future research directions include investigating the impact of physical conditions, such as height and weight, on deep-learning model results. Additionally, the study aims to explore ways to enhance breathability and user comfort while ensuring optimal device adherence to the skin.

In summary, this article introduces a novel skin-wearable motion monitoring device

for accurate and comfortable fall detection in older adults. The study incorporates deep learning algorithms, explores optimal device placement, and emphasizes the importance of collecting motion data directly from older adults. The findings contribute to improving fall detection systems tailored to the specific needs of the older adult population.

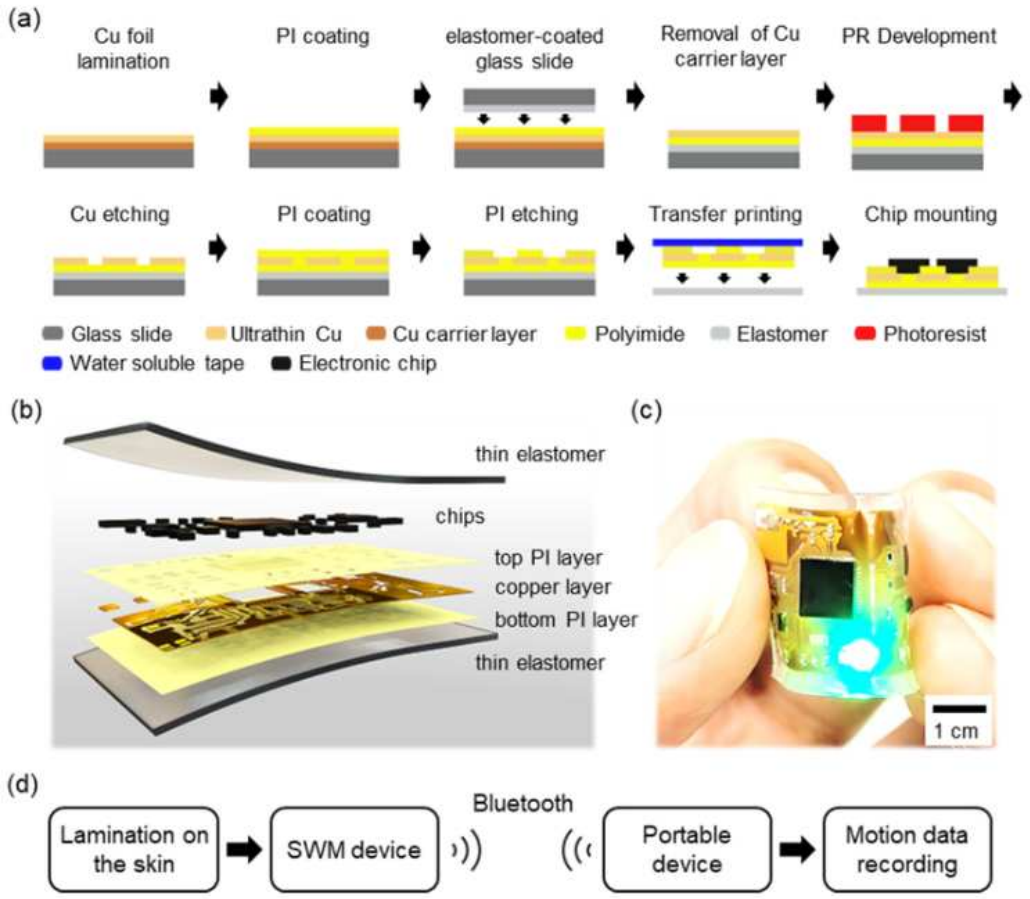

Figure 3.2: The overview of the skin-wearable device. (a) Illustration of device fabrication process using an ultrathin Cu film. (b) Exploded view of the device showing materials for each layer. (c) Device-bending. (d) The flow of motion data from the skin to analysis. used by Lee Y et al. (2023) [43].

3.3.3 Al-qaness M et al. (2022)

The reviewed paper [37] focuses on the application of metaheuristic (MH) optimization algorithms in human activity recognition (HAR) and fall detection using sensor data. The paper highlights the potential of MH algorithms in complex engineering and optimization problems, particularly in feature selection (FS).

To improve the classification accuracy of HAR and fall detection, the paper utilizes nine MH algorithms as FS methods. These algorithms include Aquila optimizer (AO), arithmetic optimization algorithm (AOA), marine predators algorithm (MPA),

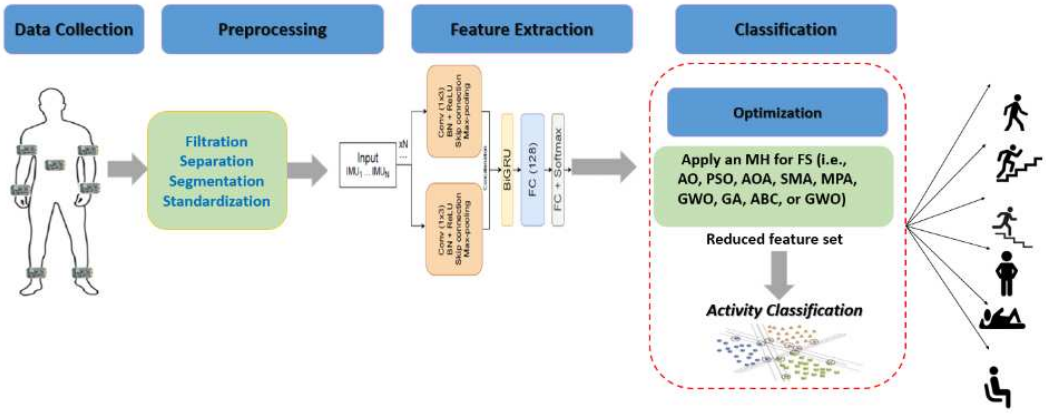

Figure 3.3: The main workflow by Al-qaness M et al. (2022) of HAR application using the integration of deep learning and MH optimization algorithms [37].

artificial bee colony (ABC) algorithm, genetic algorithm (GA), slime mold algorithm (SMA), grey wolf optimizer (GWO), whale optimization algorithm (WOA), and particle swarm optimization algorithm (PSO).

The paper proposes a comprehensive approach that involves efficient preprocessing and segmentation methods to identify motion patterns and reduce time complexities. It also introduces a novel feature extraction technique based on ResRNN, a deep learning model that incorporates convolutional neural networks (CNN), residual networks, and bidirectional recurrent neural networks (BiRNN).

The MH algorithms are then employed to select optimal features and enhance classification accuracy. Support vector machine (SVM) and random forest (RF) classifiers are used for multi-classification and binary classification tasks, respectively. The evaluation is performed on seven diverse datasets: PAMMP2, Sis-Fall, UniMiB SHAR, OPPORTUNITY, WISDM, UCI-HAR, and KU-HAR.

The results demonstrate the promising performance of MH optimization algorithms in HAR and fall detection applications. The paper suggests further exploration of advanced MH methods, such as modified algorithms integrated with intelligent search mechanisms like levy flight, opposition-based learning, and hybridizations of multiple MH algorithms.

In summary, the reviewed paper presents a comprehensive approach that combines deep learning and MH optimization algorithms for HAR and fall detection. The findings highlight the potential of MH algorithms in enhancing classification accuracy and open avenues for future research in sensor-based applications.

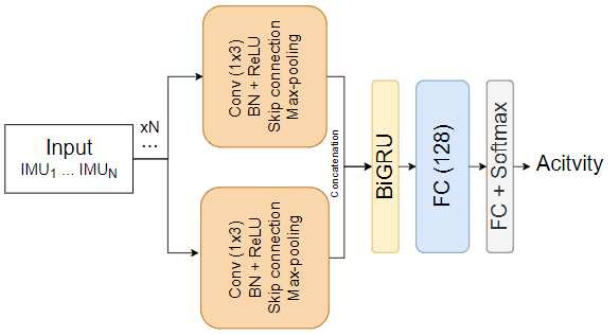

Figure 3.4: The proposed ResRNN model for feature extraction by Al-qaness M et al. (2022) [37].

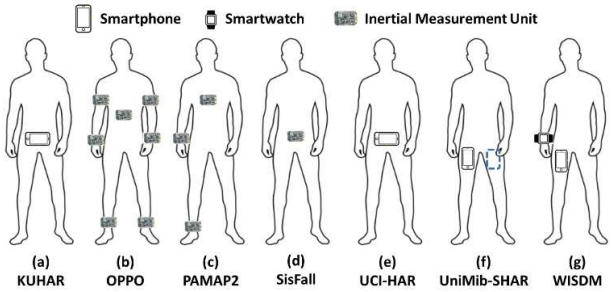

Figure 3.5: Sensor placement on the subject's body by Al-qaness M et al. (2022). A waist-mounted smartphone was used for KUHAR, UCI-HAR, and WISDOM. WISDOM was collected using additional smartwatches. IMUunits were used to collect the signals for OPPO (positions: back, right/left upper/lateral parts of the arm and right/left shoe), PAMAP2 (positions: chest, wrist, and ankle), and Sis-Fall (position: waist) [37].

3.3.4 Ruiz J et al. (2022)

This research study [46] focused on fall detection and prevention for older adults during bedtime. The proposed system Figure 3.7 aims to address the growing need for intensive care and attention due to the increasing number of people requiring assistance. Falls among older adults are a significant health concern and the second leading cause of unintentional death worldwide. However, widespread solutions for fall detection and prevention face challenges such as privacy issues, high costs, low performance, and discomfort associated with wearable devices.

The research presents a comprehensive solution that focuses on monitoring the position in bed and automatically detecting falls. By monitoring bed exits, especially for individuals with cognitive impairments or mobility issues, the system aims to identify risk situations promptly. The proposed system combines real-time monitoring of bed position with an automatic fall detection system. It offers a low-cost solution, ensuring user privacy without the need for uncomfortable devices.

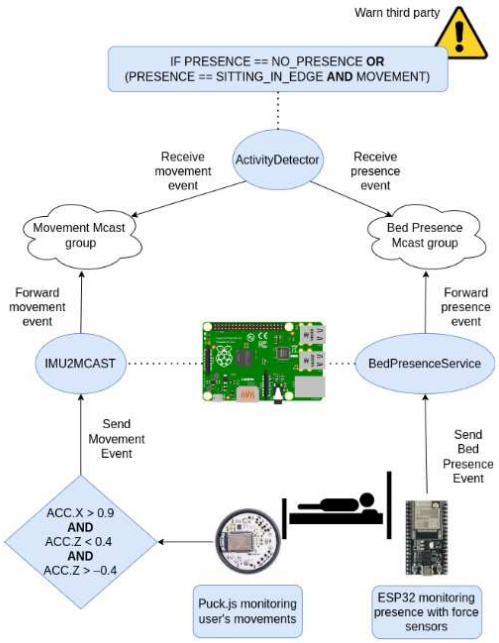

Figure 3.6: Overview of the fall-prevention system by Ruiz J et al. (2022) [46].

Experimental validation of the proposed system was conducted with young adults, yielding promising results. The fall detection system achieved an accuracy of 93.51%, sensitivity of 92.04%, and specificity of 95.45%. Risk situations, such as transitioning from lying on the bed to sitting on the bedside, were recognized with 96.60% accuracy, while bed exits were recognized with 100% accuracy.

The conclusion highlights the requirements for a widely adopted fall prevention Figure 3.6 and detection system, including privacy awareness, real-time operation, avoidance of uncomfortable devices, and affordability. The proposed system fulfills these requirements by utilizing a Raspberry Pi 4, the IMU Puck.js sensor Figure 3.8, and three pressure sensors. The installation is simple, and user privacy is ensured compared to video-based solutions. The wearable sensors are discreet and comfortable for the user.

Future work in fall detection and prevention will focus on improving the accuracy of the system, particularly in increasing sensitivity. This can be achieved through the implementation of a neural network for classification algorithms and gathering more fall data to enhance the model. Additionally, extending the monitoring range beyond a single room will be explored, either by utilizing WiFi modules for cloud processing or processing data at the edge and communicating results to a gateway.

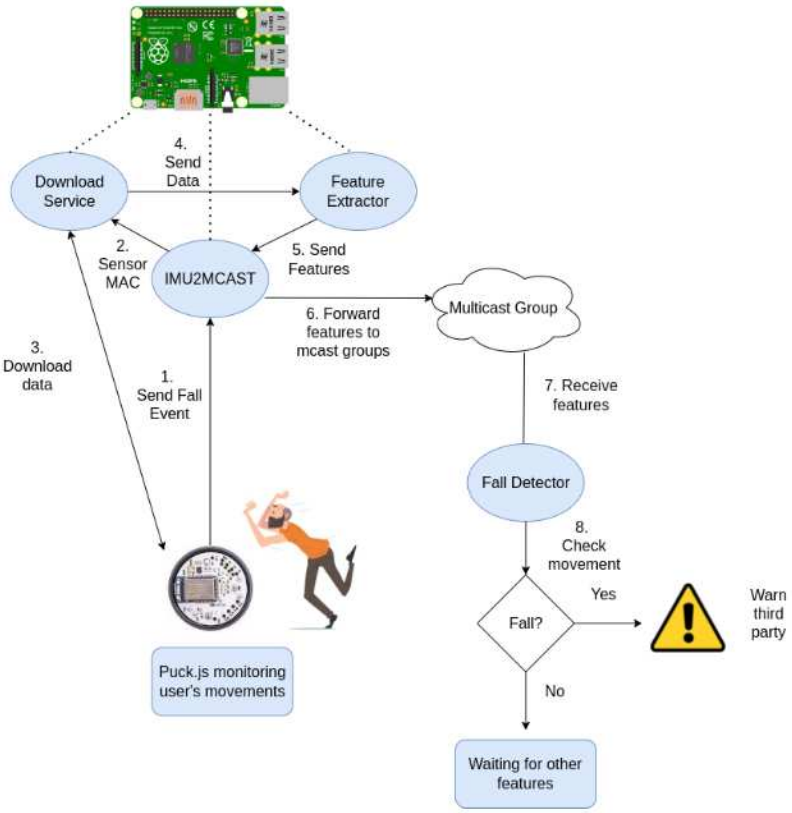

Figure 3.7: Proposed system architecture by Ruiz J et al. (2022) [46].

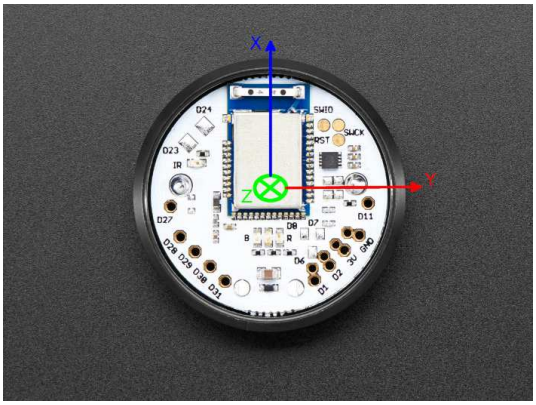

Figure 3.8: IMU sensor used for this work with its respective axis by Ruiz J et al. (2022) [46].

For fall prevention, future work will concentrate on improving bed position detection to enhance the accuracy of detecting the intention to exit the bed. This improvement will undergo further testing to identify any issues with sensors and explore advanced techniques for bed position classification. Monitoring bed positions can also enable additional functionalities, such as tracking postural changes in nursing homes and measuring the quality of sleep based on individual movements throughout the night.

Overall, the research study contributes to the development of a comprehensive, cost-effective, and privacy-conscious system for fall detection and prevention during bedtime. The proposed system's experimental validation demonstrates its reliability and potential impact on older adults and caregivers.

3.3.5 Mankodiya H et al. (2022)

This research study [38] focuses on developing a fall detection system Figure 3.9 designed to ensure the safety of older individuals. The system uses multiple sensors placed at different locations on the body to gather data for training. The goal is to overcome the limitations of previous approaches, which either relied on a single sensor or were confined to a specific area. The proposed system achieves an accuracy ranging from 93.1% to 97.2% for individual sensor models, with an overall accuracy of 92.54% when using a majority voting classifier.

To enhance the interpretability of the models, an explainable artificial intelligence (XAI) technique called LIME (Local Interpretable Model-Agnostic Explanations) Figure 3.10 is incorporated. LIME helps to provide insights into the model's outputs and decision-making process. The authors use LIME to generate explanations and analyze the correlations between the model's predictions and real-life scenarios. They present LIME graphs for activities of daily living (ADL) and falls, highlighting distinct decision-making patterns in the prediction models. Specifically, negative LIME values are prevalent for ADL predictions, while positive values dominate for fall predictions.

Although the explanations proposed in the paper offer specific insights into the model's predictions, the authors acknowledge that they do not entirely address the black-box issue associated with modern machine-learning algorithms.

In terms of future work, the authors suggest optimizing the system by reducing the time complexity of the prediction models. This optimization would involve allowing variable length timestamps for data input streams and adjusting predictions accordingly. Additionally, utilizing variable length timestamps would enable the LIME-based explainability module to dynamically present explanations, leading to reduced inference time for the explanations.

In summary, the mentioned work focuses on developing a robust fall detection system using multiple sensors and achieving high accuracy. It incorporates an XAI technique (LIME) to enhance interpretability and provides insights into the decision-making process of the models. The future work involves optimizing the system's time complexity and inference time for explanations by adopting variable length timestamps.

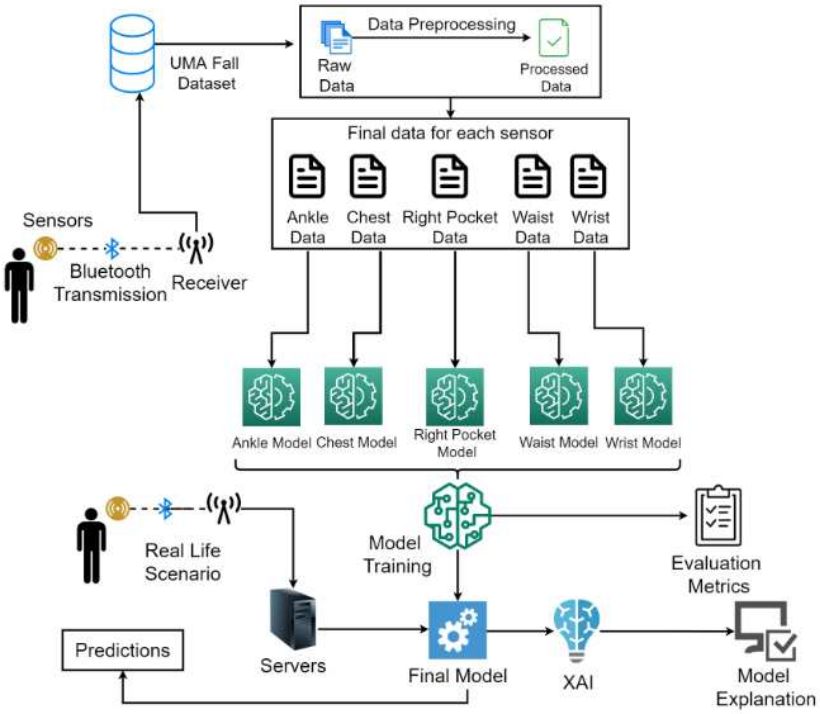

Figure 3.9: Proposed system for fall detection using wearable technology by Mankodiya H et al. (2022) [38].

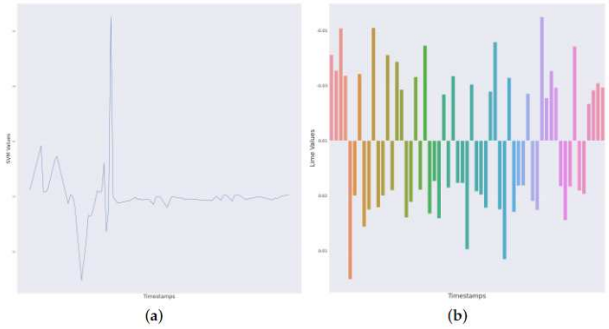

Figure 3.10: Original activity class: fall, Predicted activity class: fall, Prediction probability: 0.94. The pair of figures shows SMV values from the ankle sensor and LIME values generated on the trained model. The timestamp axis, i.e., the x-axis, should be considered common for both the above sub-plots. (a) Lineplot of SMV values by accelerometer for sensor ankle for the given timestamps. (b) Barplot for LIME values for sensor ankle for the given timestamps Mankodiya H et al. (2022) [38].

3.3.6 Musci M et al. (2021)

This research [39] focuses on the feasibility and effectiveness of using Deep Learning (DL) techniques, specifically Recurrent Neural Networks (RNNs) based on Long Short-Term Memory (LSTM) cells Figure 3.11, for fall detection in wearable devices. The objective of Fall Detection Systems (FDSs) is to detect falls as soon as they occur and automatically request assistance to prevent severe injuries or death.

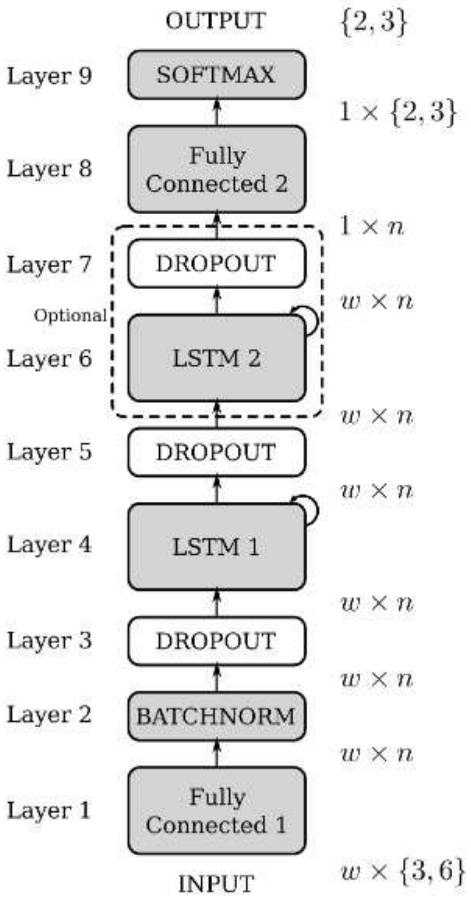

Figure 3.11: The proposed RNN architecture. White blocks are active during the training phase only and are not implemented in the run-time module. The input size is determined by the number of sensors (accelerometers and/or gyroscopes), the sensory input dimensions, and the size of the sliding window; which represents the cell's inner dimension. The output size depends on the number of classes used in annotating the dataset, namely two (BKG and FALL) or three (BKG, FALL, and ALERT). Layers 6 and 7 can be replicated up to k times by Musci M et al. (2021) [39].

The paper emphasizes the advantages of wearable embedded sensors for 24/7 monitoring, as they are effective, less intrusive, and cost-effective compared to other systems. The study focuses on designing a DL model suitable for onboard wearable

devices, taking into account the limited memory, computation resources, and power consumption of microcontrollers.

The authors demonstrate that a minimal RNN architecture based on LSTM cells achieves comparable performance to state-of-the-art fall detection systems. They extended the publicly available SisFall dataset by adding temporal annotations Figure 3.12 to mark the actual occurrences of falls, addressing the need for appropriate datasets with temporal annotations for training RNNs.

Figure 3.12: iThe annotation tool used to enhance the SisFall dataset. The top pane shows an example sequence of accelerometer readings (in g) corresponding to a backward fall caused by a slip. The blue interval marks a FALL event while the orange interval corresponds to an ALERT event. The bottom pane shows the temporal labeling produced as output, which becomes part of the dataset enhancement by Musci M et al. (2021) [39].

Experimental results show that the proposed LSTM-based architecture, trained on the extended SisFall dataset, achieves accuracy levels exceeding 96% for fall detection. The feasibility of implementing the runtime module on a real hardware device, specifically the SensorTileR© board, is also validated. This implementation demonstrates the viability of the proposed approach for real-time fall detection with minimal power consumption.

The study suggests a design strategy for DL methods in fall detection, emphasizing the importance of finding the simplest network architecture that meets the desired task while requiring limited computation power and memory. Furthermore, the authors discuss the need for complete and extensive datasets with appropriate annotations. They propose collecting a new dataset with wearable sensors based on SensorTileR© and associated video recordings to improve the accuracy of temporal interval identification.

In terms of future work, the authors suggest exploring unsupervised DL approaches for fall detection, as well as conducting tests with real subjects in real scenarios. These directions could further enhance the embedded implementation of fall detection systems for wearable devices.

3.3.7 Casilari E et al. (2020)

This study [44] explores the effectiveness of using a gyroscope and an accelerometer, combined with a deep learning strategy, to discriminate falls from ordinary Activities of Daily Living (ADLs).

The study employs a Convolutional Neural Network (CNN), a deep learning architecture, to directly extract features from raw data collected by the sensors, eliminating the need for complex preprocessing and manual feature engineering. A well-known public dataset containing a large number of mobility traces, including falls and ADLs, is utilized to evaluate the performance of the CNN when fed with signals from both the gyroscope and accelerometer. The CNN is carefully tailored and hyper-parameterized to optimize the classification performance.

The experiments reveal that the configured CNN achieves better results when the gyroscope measurements are disregarded, and the accelerometer signals alone are used as input features for training and testing the convolutional neural classifier. Surprisingly, the exclusion of gyroscope data leads to improved classification performance. This finding challenges the prevailing assumption that combining gyroscope and accelerometer measurements enhance fall detection accuracy.

The study's outcomes indicate that a deep learning classifier can effectively characterize user mobility by solely focusing on the raw accelerometer signals. This finding suggests that the extra costs associated with gyroscope utilization, such as energy consumption, hardware complexity, and processing requirements, may not be necessary. Additionally, by relying exclusively on accelerometer data, the dimensionality of the detection algorithm's input features can be reduced, enabling real-time operation on wearable devices with limited computing resources. It is important to note that further research using additional public datasets should be conducted to confirm these conclusions.

In conclusion, This study contributes to the literature by evaluating the impact of gyroscope and accelerometer data on the performance of fall detection systems in wearable devices. Contrary to previous assumptions, the results demonstrate that the exclusive use of accelerometer signals, without incorporating gyroscope data, yields superior classification results. These findings have implications for the design and implementation of wearable FDSs, emphasizing the potential of deep learning architectures in directly processing raw accelerometer data for effective fall detection.

3.3.8 Liu L et al. (2020)

This research [40] provides an overview of a novel low-power fall detection workflow for the elderly, as presented in the referenced paper. The primary objective of this research is to develop an accurate and energy-efficient fall detection module to enhance community-based care for the elderly, thereby reducing health risks. The proposed workflow consists of two main components: a sensing module-integrated energy-efficient sensor and a server-side deep neural network for fall detection (FD-DNN) Figure 3.13.

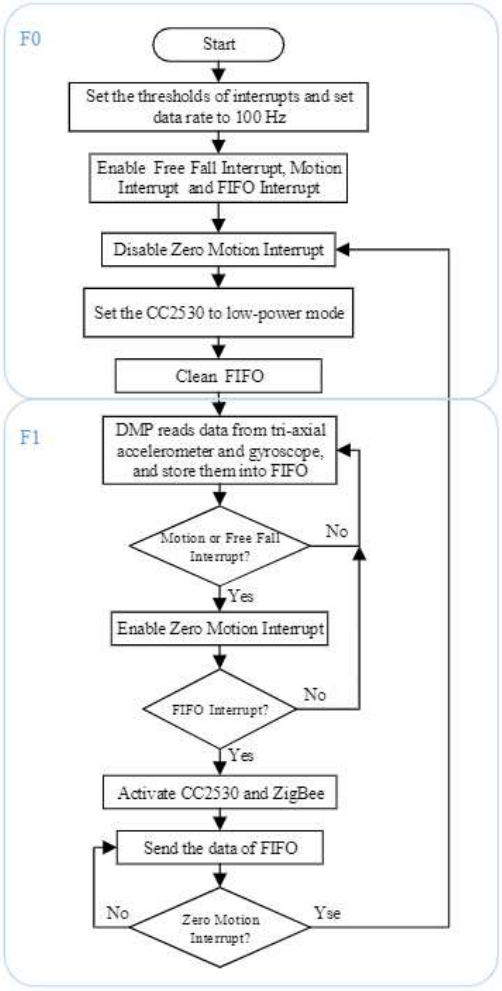

Figure 3.13: The flowchart of the data sensing and transmission algorithm by Liu L et al. (2020) [40].

The sensing module-integrated energy-efficient sensor is designed to sense and cache human activity data in sleep mode, thereby conserving energy. An interrupt-driven algorithm is proposed to transmit the cached data to a server integrated with ZigBee technology. This approach ensures accurate detection of falls while maintaining energy efficiency.

The server-side FD-DNN is carefully designed using a combination of convolutional neural networks (CNN) and long short-term memory (LSTM) algorithms Figure 3.14. The FD-DNN is trained and tested on both online and offline datasets to evaluate its performance. Experimental results demonstrate that the FD-DNN achieves a fall detection accuracy of 99.17%, with a specificity of 99.94% and a sensitivity of 94.09%. Additionally, the FD-DNN exhibits low power consumption characteristics, making it suitable for practical fall detection applications.

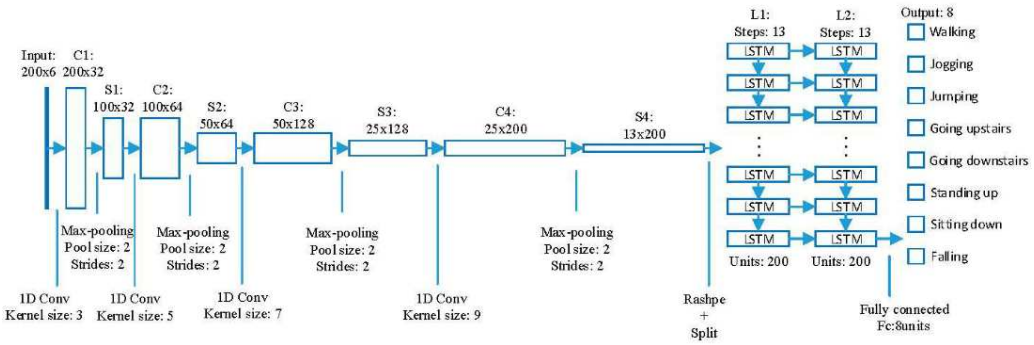

Figure 3.14: The architecture of the FD-DNN by Liu L et al. (2020) [40].

The conclusion of the paper highlights the significance of the proposed low-power fall detection workflow. The interrupt-driven low-power motion sensing module effectively senses and transmits human activity data, while the server-side FD-DNN accurately distinguishes falls from activities of daily living (ADLs). The FD-DNN outperforms traditional classification algorithms in terms of fall detection accuracy. Furthermore, the proposed workflow achieves a battery life of more than 140 hours in real-life scenarios Figure 3.15, further emphasizing its practicality for fall detection in the elderly. The future work suggested in the paper aims to simplify the network and improve the speed of fall detection.

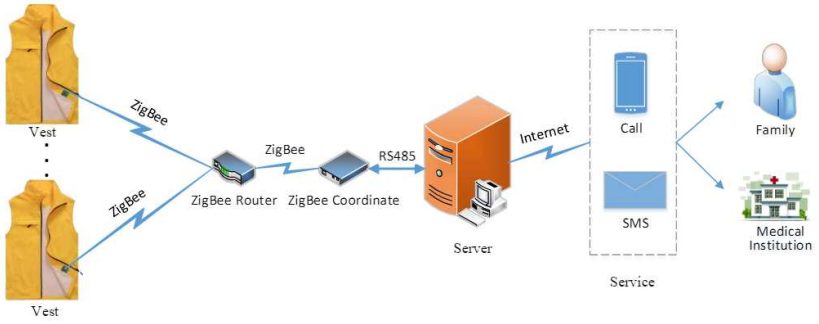

Figure 3.15: The experimental environment for fall detection by Liu L et al. (2020) [40].

In summary, this research provides an overview of a novel low-power fall detection workflow for the elderly. The paper's contributions include the design of an energy-

efficient sensing module, the development of an interrupt-driven algorithm for data transmission, and the implementation of a server-side FD-DNN utilizing CNN and LSTM algorithms. The experimental results demonstrate the effectiveness and accuracy of the proposed workflow in distinguishing falls from ADLs, while also highlighting its low power consumption characteristics.

3.3.9 Kim T et al. (2020)

This study [41] proposed a novel method for predicting impact acceleration magnitude prior to a fall using a single inertial measurement unit (IMU) sensor and a sequential-based deep learning model Figure 3.16. The goal was to develop a fall prevention intervention, such as a wearable hip airbag system, by optimizing the deployment process to minimize fall-related injuries in real-time.

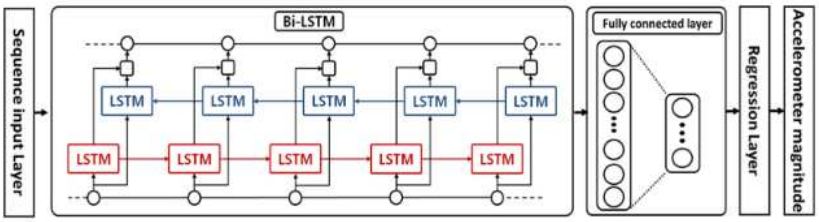

Figure 3.16: Detail architecture of long short-term memory network for fall accelerometer magnitude prediction by Kim T et al. (2020) [41].

Twenty-four healthy participants took part in fall experiments, wearing a single IMU sensor on their waist to collect tri-axial accelerometer and angular velocity data. The researchers employed a bi-directional long short-term memory (LSTM) regression model for predicting the impact acceleration magnitude before a fall occurred in five different directions.

To improve the prediction performance, the study utilized data augmentation techniques, including jittering, scaling, and time-warping, to increase the robustness of the training model and enhance the temporal characteristics of the sensor data. The results demonstrated that applying all three data augmentation techniques yielded a mean absolute percentage error (MAPE) of 6.69±0.33% and an r-value of 0.93. Moreover, increasing the number of training datasets by four-fold resulted in a significant reduction of MAPE by 45.2%.

The discussion highlighted the significance of the study, emphasizing that it is the first to develop a model using deep learning and wearable sensor signals to predict impact acceleration magnitude. The synthetic training data generated through data augmentation helped mitigate overfitting, enhance the generalizability of the model to unseen data, and address biases caused by imbalanced training data volume. The study's results showed that employing data augmentation techniques improved the performance of predicting fall impact acceleration magnitude compared to not using

augmentation. Furthermore, increasing the number of datasets and applying data augmentation demonstrated a consistent decrease in MAPE Figure 3.17, aligning with findings from previous studies.

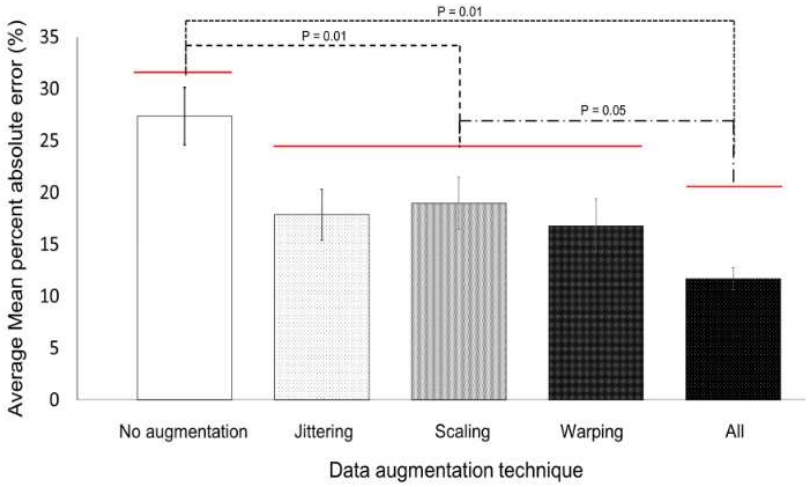

Figure 3.17: Average mean absolute percent error of fall impact acceleration magnitude prediction for three different data-augmentation techniques by Kim T et al. (2020) [41].

Overall, this study contributes to the field of fall prevention by proposing a reliable method for predicting impact acceleration magnitude Figure 3.18 using a single IMU sensor and deep learning. The results offer valuable insights for the design and development of fall prevention interventions, such as wearable airbag systems, and can inform the optimization of parameters like pressure or air flow rate for deploying airbags to minimize fall-related injuries in real-time.

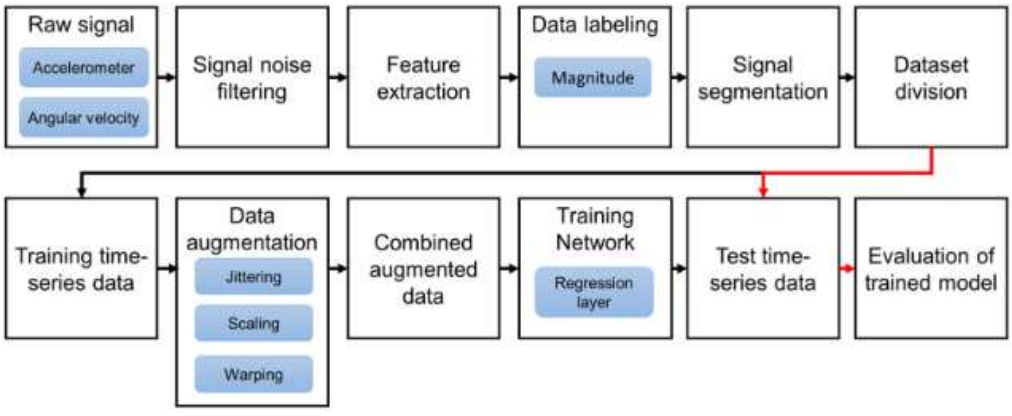

Figure 3.18: Overall architecture for predicting impact acceleration magnitude during fall by Kim T et al. (2020) [41].

3.3.10 Hussain F et al. (2019)

The prevalence of falls among older adults is a significant concern due to the associated severe injuries, disabilities, and even fatalities. Detecting falls and understanding the activities leading to them is crucial for preventing future incidents. Wearable sensors and systems have emerged as a promising approach for the continuous monitoring of elderly individuals. While most existing wearable fall monitoring systems focus solely on fall detection, it is equally important to identify the cause of the fall incident.

In this context, this research paper [42] proposes a wearable sensor-based continuous fall monitoring system that not only detects falls but also recognizes the falling pattern and associated activities Figure 3.19. The proposed methodology combines conventional signal processing techniques with machine learning algorithms. Real-time fall monitoring involves continuously acquiring sensor data in ten-second chunks using a non-overlapping window. The acquired data is processed to determine the occurrence of a fall. If a fall is detected, the system further processes the data to identify the falling activity using supervised learning.

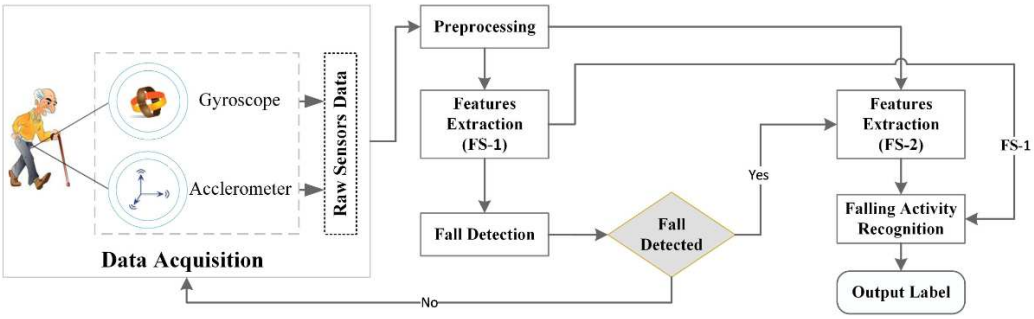

Figure 3.19: Proposed methodology for fall detection and falling activity recognition (FAR) by Hussain F et al. (2019) [42]

The proposed methodology consists of five major stages: data acquisition, pre-processing, feature extraction, fall detection, and falling activity recognition. The performance of the system is evaluated using three machine learning algorithms: k-nearest neighbors (KNN), support vector machine (SVM), and random forest (RF). The results demonstrate high accuracy in fall detection, with 99.80% achieved using the KNN classifier. For recognizing different falling activities, the highest accuracy achieved is 96.82% using the RF classifier.

The study emphasizes the importance of being aware of daily living activities associated with falls alongside fall detection. Existing research in fall detection has focused on distinguishing falling activity from non-falling activity. However, incorporating knowledge about the specific activity leading to a fall can significantly enhance the effectiveness of fall prevention strategies. The proposed methodology utilizes a publicly available SisFall dataset and employs accelerometer and gyroscope sensors for fall detection and falling activity recognition.

The findings highlight the effectiveness of the proposed methodology in accurately detecting falls and recognizing various falling activities. The combination of accelerometer and gyroscope data improves recognition performance, indicating the value of using multiple sensors. Certain falling activities, such as falling backward while walking, falling forward while jogging, falling backward when trying to sit down, and falling forward while sitting, achieve recognition rates exceeding 90%.

Future research directions include extending the proposed solution to incorporate fall prediction and prevention through gait and activity pattern analysis. Additionally, exploring the relationship between different types of falls and activities of daily living associated with them can provide valuable insights. Integration of additional sensing modalities, such as cameras and pressure sensors, may further enhance fall detection and recognition accuracy.

In summary, this research paper presents a comprehensive wearable sensor-based fall monitoring system that goes beyond fall detection by identifying falling patterns and associated activities. The proposed methodology achieves high accuracy in both fall detection and falling activity recognition. The findings contribute to the advancement of fall prevention strategies and lay the foundation for further exploration in predicting falls and incorporating complex falling activities into the system.

3.3.11 Shahzad A et al. (2019)

This research [36] discusses the development of an Android-based fall detection system called FallDroid Figure 3.20. The system aims to address the challenges posed by falls in the elderly population and improve their functional ability and confidence level. FallDroid utilizes the embedded accelerometer sensor in smartphones to monitor and detect fall events through a two-step algorithm.

The first step of the algorithm employs a threshold-based method (TBM) to effectively filter out most activities of daily living (ADL) data, focusing on fall-like events. However, TBM techniques have limitations in simultaneously avoiding false negatives (missed falls) and false positives (ADL classified as falls). Fine-tuning the thresholds poses a tradeoff between the two, making it challenging to achieve optimal thresholds for consistent performance across different individuals.

To overcome this limitation, the second step of the algorithm utilizes a pattern recognition technique called multiple kernel learning support vector machine (MKL-SVM) Figure 3.21. This step helps classify difficult fall-like events and reduces false alarms while maintaining low computation cost and power consumption.

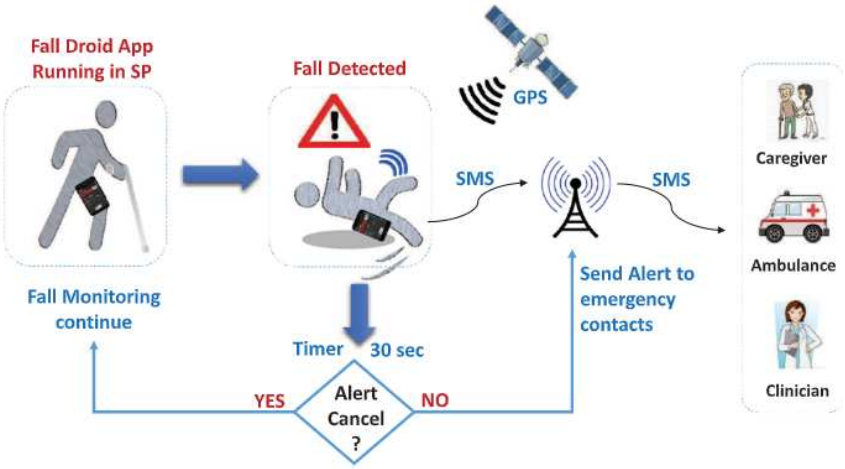

Figure 3.20: Overview of the fall detection and emergency alert system by Shahzad A et al. (2019) [36].

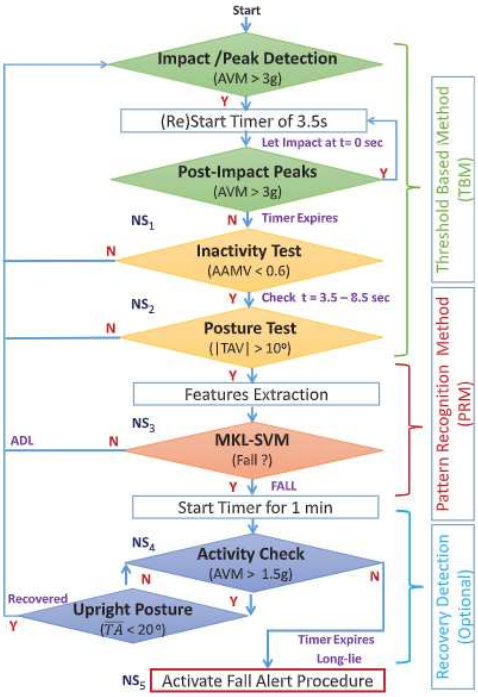

Figure 3.21: Fall detection algorithm (proposed), comprising of two main steps: TBM and pattern recognition method, with an optional recovery detection part by Shahzad A et al. (2019) [36].

Experimental results demonstrate that FallDroid achieves high accuracy, sensitivity, and specificity when placed around the waist and thigh. It outperforms existing applications in both offline and online analysis. The system also exhibits a low false alarm rate, making it highly reliable for fall detection.

In future work, the authors suggest testing FallDroid in routine usage involving the elderly population, which would provide valuable data for further refinement of the algorithm. Additionally, they propose the possibility of developing a personalized user-dependent fall detection system by training the MKL-SVM classifier with individual user data. Moreover, enhancing the overall system performance could involve exploring feature engineering and extraction techniques, such as considering higher-order derivatives or nonlinear features, especially in the TBM part of the algorithm, to increase specificity.

In summary, FallDroid presents a pervasive fall detection system using smartphones that effectively detects falls in the elderly population. Its two-step algorithm, combining TBM and MKL-SVM, demonstrates promising results in terms of accuracy, sensitivity, and specificity. The system's low power consumption and user-friendly GUI make it suitable for practical implementation. However, further research is suggested to validate the system's performance in real-world scenarios and explore potential enhancements in feature engineering.

3.3.12 Luna-Perejón F et al. (2019)

The study [16] evaluated the effectiveness of four different RNN architectures using the SisFall dataset at varying frequencies. The models were integrated into two distinct embedded systems to assess execution times and changes in model performance. Additionally, a power consumption analysis was conducted. The results demonstrated a sensitivity of 88.2% and specificity of 96.4%. The simplest models achieved inference times under 34 ms, enabling real-time fall event detection with energy efficiency. These findings suggest that RNN models can be effectively implemented on low-power microcontrollers to create autonomous wearable fall detection systems in real-time.

The research was based on the SisFall dataset, encompassing simulated activities categorized as falls and Activities of Daily Living (ADL). The dataset consisted of accelerometer and gyroscope measurements captured at 200 Hz from waist-worn devices. Notably, the dataset included fall hazard events, moments preceding a fall or situations where falls were averted. The data was segmented into blocks of 256 samples (1.28 seconds) with a 50% overlap. Additionally, reduced-sample versions were created to simulate lower sampling rates Figure 3.22.

To enhance performance, batch normalization was applied, and different RNN architectures were explored 3.23. These architectures incorporated LSTM or GRU layers, with more complex versions featuring an additional RNN layer of the same type. Model hyperparameters such as batch size and learning rate were optimized through grid search, and dropout techniques were employed for improved generalization.

Two STM32 32-bit microcontrollers were chosen for integrating and analyzing the trained models. These microcontrollers were based on ARM Cortex-M4 processors, offering real-time capabilities, digital signal processing, and low-power operation.

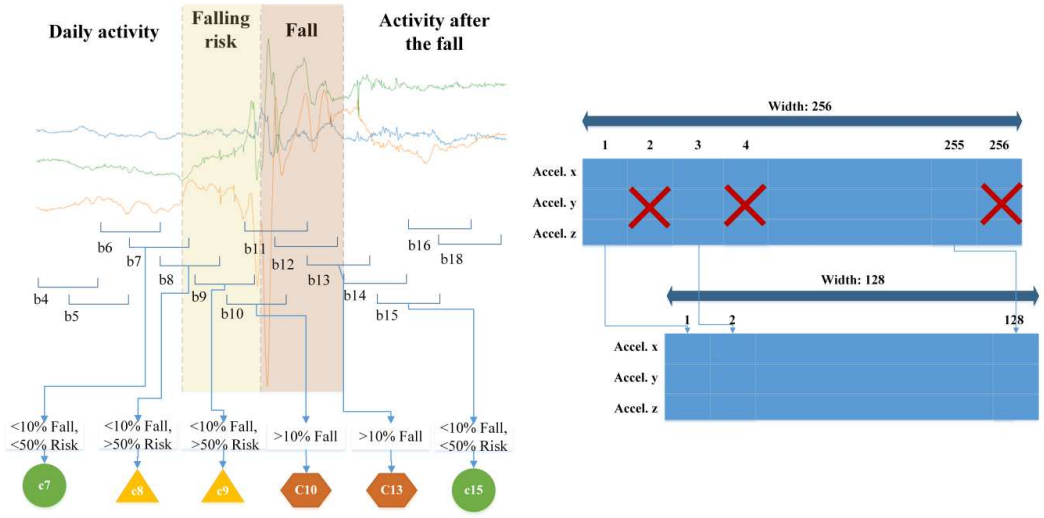

Figure 3.22: On the left: Recording segmentation and labeling process. Green circles, yellow triangles, and red hexagons indicate the block is classified as a background (BKG), a risk of falling (ALERT), or a fall event (FALL), respectively. On the right: Block width reduction process. In the case illustrated a 256-width block, corresponding to a frequency sampling of 200 Hz, is reduced to 128 samples to obtain a frequency sampling of 100 Hz. The same process was performed with 128-width and 64-width blocks to obtain 64-width (50 Hz) and 32-width (25 Hz) datasets, respectively by Luna-Perejón F et al. (2019) [16].

The study demonstrated the feasibility of real-time wearable fall detection systems using RNN architectures. Even with reduced sampling frequency, model effectiveness remained largely unaffected. Estimated power consumption indicated the feasibility of using small batteries, enabling the design of user-friendly, miniature devices. While sensitivity was slightly lower compared to other classifiers, accuracy, and specificity were higher or comparable Figure 3.24.

The work encourages further exploration of deep learning RNN architectures for wearable fall detection systems. Future research will involve extensive testing of the proposed model with new participants to validate its efficacy in real-world scenarios. In conclusion, this research signifies RNN architectures as a promising avenue for enhancing the effectiveness of wearable fall detection systems, thereby advocating for continued investigation and potential model refinement.

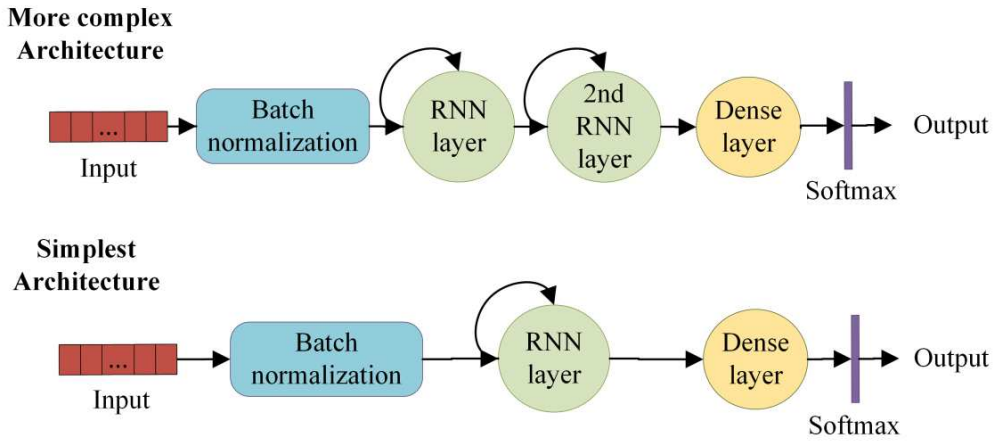

Figure 3.23: Diagram of the four recurrent neural network (RNN) architectures analyzed in this study by Luna-Perejón F et al. (2019) [16].

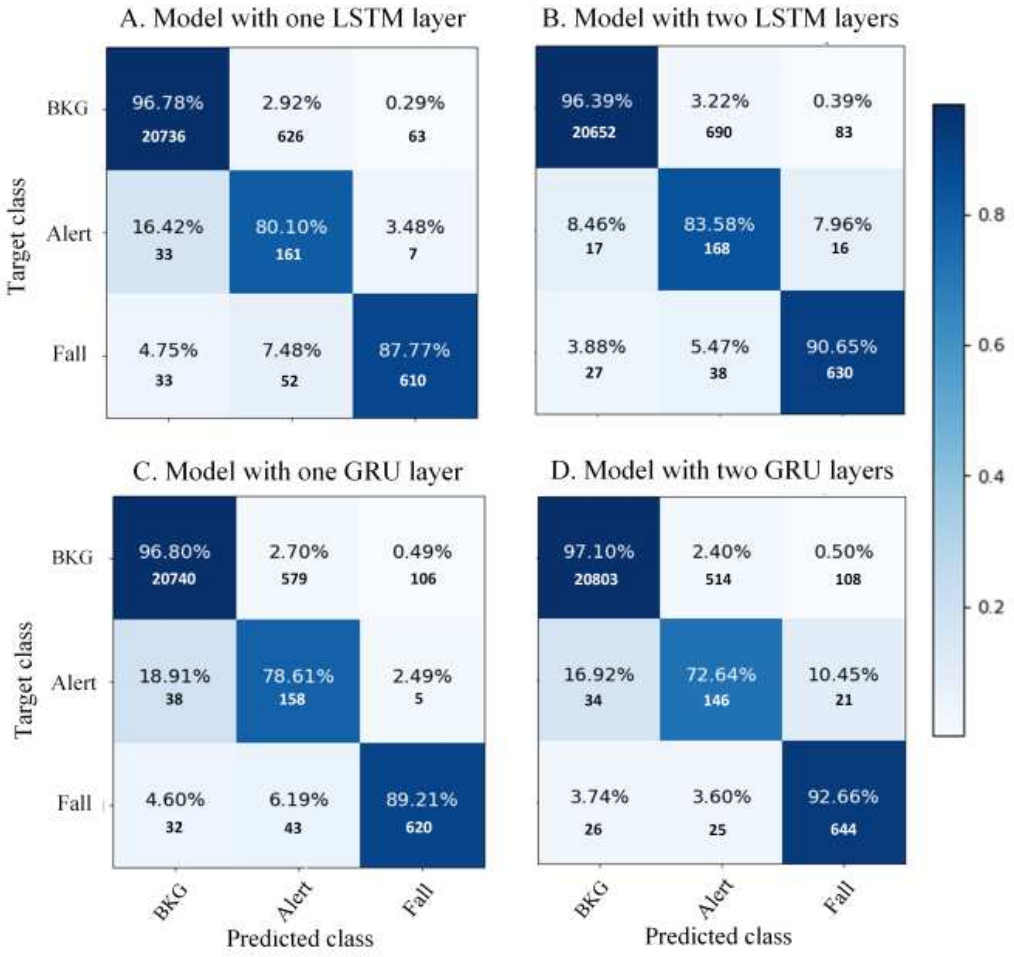

Figure 3.24: Confusion matrix of the best models for each architecture considered (at 25 Hz) by Luna-Perejón F et al. (2019) [16].

3.3.13 Ahmed M et al. (2017)

This study [47] commenced by creating a dataset named 'SMotion,' comprising specific postures recognized as potential contributors to falls in the elderly population. Employing a network of Shimmer sensors positioned on the subjects' bodies, the dataset was subsequently categorized based on age and weight groups.

The study proceeded to develop a classification system employing three distinct classifiers: support vector machine (SVM), K-nearest neighbor (KNN), and neural network (NN). This system aimed to systematically select pertinent features and accurately classify instances. Furthermore, a prototype was engineered to generate alerts directed at caregivers, healthcare professionals, or emergency services in the occurrence of a fall event.

Upon evaluating the system's efficacy, SVM, KNN, and NN classifiers were applied. Notably, the analysis revealed KNN to be the most precise classifier, achieving an impressive accuracy of 96% for age group classification and 93% for weight group classification.

In conclusion, the study introduces a fall detection system rooted in classification techniques, tailored for elderly individuals. Leveraging a network of Shimmer sensors, the system exhibited the capability to identify potential falls. The SMotion dataset was meticulously categorized by age and weight groups to enhance the precision of the analysis. The proposed system's effectiveness was substantiated by favorable performance outcomes across various scenarios.

The study's assumptions included focusing on elderly individuals without chronic health conditions, isolating specific activities of daily living (ADL) that could precipitate falls, operating within a controlled environment to minimize external influences, and categorizing data based on age and weight groups to optimize accuracy.

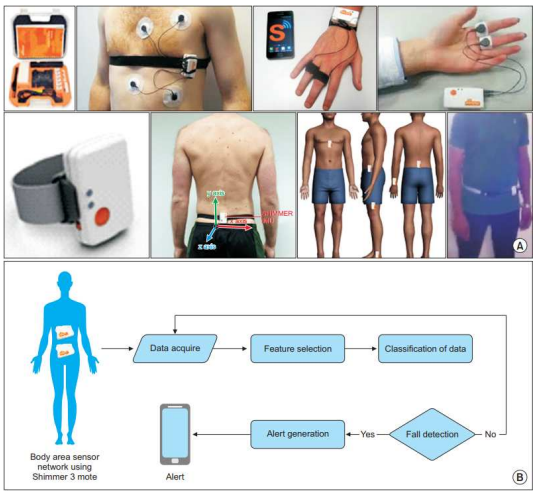

Figure 3.25: Using Shimmer for fall detection. (A) Deployment of Shimmer kit. (B) Proposed system architecture by Ahmed M et al (2017) [47].

3.3.14 Hsieh C et al. (2017)

The researchers in [18] present a novel hierarchical fall detection algorithm Figure 3.26 that combines threshold-based and knowledge-based strategies to accurately identify fall events. The algorithm addresses the challenges of fall detection through a multi-phase approach, encompassing free fall, impact, and rest phases, and offers a comprehensive evaluation involving seven types of falls and daily activities.

The threshold-based classification method effectively discerns fall events from continuous sensor data and distinguishes between absolute falls and activities of daily living (ADLs). To enhance classification accuracy, essential features are extracted from the signals, including time-domain statistical measures such as mean, standard deviation, variance, and more. A total of 54 features, derived from different axes of acceleration data, are employed in both the training and testing phases.

The knowledge-based approach employs a multi-class SVM technique to classify fall events into their distinct phases—free fall, impact, and rest. The threshold-based classification serves to differentiate between absolute falls and ADLs, reducing computational complexity for the knowledge-based algorithm. However, a significant proportion of data remains unidentified following the initial threshold-based classification, which the knowledge-based algorithm subsequently processes.

Comparative evaluations underscore the superiority of the knowledge-based approach over a machine learning-based one. The knowledge-based algorithm consistently outperforms the machine learning-based approach in terms of sensitivity, specificity, precision, and accuracy. The former achieves an overall performance of 99.79%, 98.74%, 99.05%, and 99.33%, respectively. Notably, the knowledge-based algorithm demonstrates lower standard deviations and consistently improved performance across evaluation rounds.

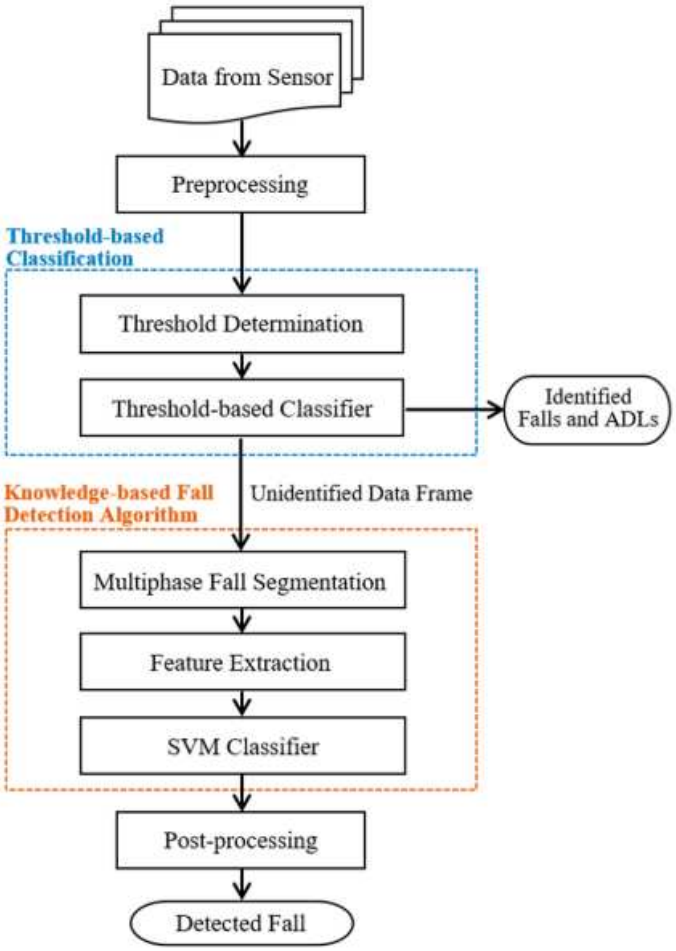

Figure 3.26: The functional diagram of the proposed fall detection algorithm by Hsieh C et al (2017) [18].

3.4 Comparison tables

Table 3.1: Study characteristics of the 14 articles included in the review

Study, yr	Population group, N(age)(gender)	Sensors	Hardware	Methods	Sensor position	Training Classes	Dataset	Dataset availability	Observation window	Data split	Performance matrics	Hyper parameters	Other
Lee Y et al., 2023	Young group, 15 (21-30 years)(-) Older adult group, 5 (>65 years)(60% females - 40% males)	XYZ acceleration and gyroscope, 6 attributes	mobile Android device	Long Short-Term Memory (LSTM)	Chest, Wrist, Necklace	Walking, stairs, sitting, running, falling (forward, backward, and lateral falls)	Collected motion data from a skin wearable device	publicly available	2sec Frames	Train (80%), test (20%)	Accuracy, precision, recall, F1-score	No. of LSTM layers: 2, Optimizer: Adam, Initial learning rate: $1x10^{-4}$, Loss function: sparse categorical cross-entropy, Intermediate layer's activation function: ReLU, Final layer's activation function: Softmax, Regularization method: Dropout, No. of trainable parameters: 475,013	Chest, Wrist, and Necklace sensors were used for the Young group, and only the Chest sensor was used for the Older group.
Marques J et al., 2023	Young group, 14 (20-46 years)(5 female - 9 male) elder group, 11 (77-95 years)(7 female - 4 males)	XYZ acceleration Fs = 40Hz	Smartwatch	KNN algorithm with euclidean distance	Wrist	11 ADLs, and 8 falls such as (Forward fall, lateral fall, backward fall) and ADLs (Walking, jogging, stairs)	Collected using Smartwatch	Not available	9sec Frames	/	Accuracy, Sensitivity, Specificity	KNN classifier based on euclidean distance with k = 3	different window sizes, sensor combinations, frequencies, and machine learning algorithms has been tested and the best configuration in term of accuracy has been reported in the table
Al-qaness M et al., 2022	Young people, 23(-)(12 female - 11 male). Elderly users, 15(-)(7 female - 8 male)	2 accelerometers (MMA8451Q and ADXL345) and one ITG3200 gyroscope	/	ResRNN with bidirectional recurrent neural network (BiRNN) for feature extraction and metaheuristics optimization algorithm(MH) for selecting optimal FS and Random forest for binary classification	Waist (SiS-FallB)	19 ADLs and 15 fall actions such as fall forward, fall backward, lateral fall, fall while jogging, fall while getting up, fall while sitting down, and others	SiS-FallB dataset	publicly available	3sec Frames	10-fold cross-validation	Accuracy, precision, recall, F1-score	Activation function: Rectified Linear Unit (ReLU), RNN structure: Gated Recurrent Units (GRU), Dropout rate: 0.3, Optimizer: Adam, Learning rate: $1x10^{-4}$, Number of epochs: 350, Early stopping: Yes (based on validation loss)	feature selection method based on 9 MH optimization algorithm, the Aquila optimizer (AO) was used since it gave the highest accuracy. 7 databases namely (KU-HAR, UCI-HAR, WISDM, PAMAP2, OPPORTUNITY, SiS-Fall, and UniMiBSHAR) were analyzed but sisFall was chosen since it gave the highest accuracy with (AO)
Ruiz J et al., 2022	young adults, 17(30±8.02 years) (4 female - 13 male)	Puck.js sensor (Magnetometer + Accelerometer), reduced Fs = 50Hz	Raspberry Pi 4	threshold-based algorithm (TBA) along with a state machine devised to determine the movements that are likely to be a fall using signal vector magnitude (SVM) and support vector machine (SVM)	Waist	5 ADLs and four different types of falls (forward fall, backward fall, fall on the right side, and fall on the left side)	dataset used in the study was collected from 17 people	publicly available	1sec Frames	Traning and testing	Accuracy, Sensitivity, Specificity	/	the sensor has also (Light sensor + Temperature). If a fall is detected by the threshold method data is sent to a pre-trained model (SVM) to decide if it's a Fall or ADL. The study focuses on detecting Fall after leaving the bed.
Mankodiya H et al., 2022	17(18-55 years) (10 males - 7 females)	tri-axial (x,y,z) accelerometer, gyroscope, and magnetometer	/	Signal Magnitude Vector (SMV) calculation are fet to sequence model (LSTM) and Majority Voting Classifier (MVC) then (LIME) is applied to obtain local explanations for the outputs	Ankle, Chest, Right pocket, Waist, Wrist	12 ADLs and 3 types of falls (forward fall, backward fall, and lateral fall)	UMA Fall dataset	publicly available	350 timestamps of all Rows in the dataset	Train (70%), Test (30%)	Accuracy, Sensitivity, Specificity	Ankle model (lstm(150) + dropout(0.2) + lstm(150) + dropout(0.4) + dense(1) + Epochs(200)), Chest model (lstm(150) + dropout(0.7) + lstm(150) + dropout(0.4) + dense(1) + Epochs(100)), RightPocket model (BatchNormilization (momentum = 0.99) + dropout(0.7) + lstm(150) + dense(1) + Epochs(360)), Waist model (lstm(150) + dense(1) + Epochs(200)), Wrist model (lstm(150) + dropout(0.8) + dense(1) + Epochs(125))	The F1 score has also been plotted showing the highest value for the Waist model, but the value of the F1 score has not been explicitly reported
Musci M et al., 2021	Young people, 23(-)(12 female - 11 male). Elderly users, 15(-)(7 female - 8 male)	two tri-axial accelerometers and a gyroscope Fs=200Hz	SensorTile© board (ARM© Cortex©M4 Micro-Controller Unit (MCU))	Accelerometer and gyroscope data labeling then filtered and normalized, and classified using Long Short-Term Memory (LSTM)	Waist	19 ADLs and 15 falls	annotated SisFall dataset	publicly available	window width of 256 samples (i.e. 1,28s, Fs=200Hz)	Train (30 subjects), Test (8 subjects). training set was further divided into train and validation sets using a random 80%/20%	Accuracy, Sensitivity, Specificity	Number of Epochs: 300 Batch size: 500 Optimizer: Adam Learning rate: $2.5x10^{-3}$ L2 Regularization: $1.5x10^{-3}$ Dropout rate: 0.8 Loss function: weighted cross entropy	weighted cross-entropy loss function was used because the classes were not balanced. To avoid identity bias, each subject in the SisFall dataset was randomly assigned to either the training or test set. the output of the classifier is BKG or FALL
Casilari E et al., 2020	Young people, 23(-)(12 female - 11 male) Elderly users, 15(-)(7 female - 8 male) Age for both groups (19–75 year)	TG3200 gyroscope, 2 accelerometers (models ADXL345 and MMA8451Q) Fs=200Hz	/	Accelerometer data observation and input to a detector (CNN)	Waist	19 ADLs and 15 falls	SisFall dataset. Number of samples 4505 (ADLs=2707, Falls=1798)	publicly available	5sec Frames	Train (60%), validation (20%), Testing (20%) with 5 fold validation	Accuracy, Sensitivity, Specificity	Mini batch size: 16 training instances Training method: Stochastic Gradient Descent with Momentum Learning rate: 0.0001 Maximum number of training epochs: 5	the input to CNN was ACC+Gyr but by only using ACC we got higher accuracy and low complexity of the model. Different observation windows and CNN layers have been assessed, but only the one with the highest accuracy has been reported.

Table 3.2: Study characteristics of the 14 articles included in the review (Continue)

Study, yr	Population group, N(age)(gender)	Sensors	Hardware	Methods	Sensor position	Training Classes	Dataset	Dataset availability	Observation window	Data split	Performance matrics	Hyper parameters	Other
Liu L et al., 2020	*SisFall dataset: 38 adults (19-61+ years, mixed gender) *MobiFall dataset: 24 volunteers (22-42 years, mixed gender) *Custom dataset: 20 volunteers (24-50 years, 17 male and 3 female)	three-axial MEMS gyroscope and accelerometer (MPU6050)	CC2530 Texas Instruments CPU (used only for reading and transmitting the data to a server)	Feature extraction using (CNN) + LSTM for classification	Waist	Falling, Standing up, Walking, Jogging, Jumping, Going upstairs, Going downstairs, Sitting down	Joint-Dataset was created from SisFall and MobiFall datasets (9600 samples), (Fall = 1200, ADLs = 8400)	Not available	2sec Frames	Train (70%), validation (10%), testing (20%)	Accuracy, Sensitivity, Specificity, Test Time	'Mini-batch size: 128 Optimizer: Adam Step size: 200 Learning rate: $1x10^{-3}$ Number of epochs: 300 Biases of the forget gate set to 1.0 in LSTM layers Dropout probability: 0.5	/
Kim T et al., 2020	24(22-34 years) (14 males - 10 females)	tri-axial accelerometer and gyroscope data (mpu6050)	TM32F103CB (used only for reading and transmitting the data to a workstation)	data augmentation and Feature extraction and filtering then Bi-Directional Long Short-Term Memory was used	Waist	Front Fall, Back Fall, Left Fall, Right Fall, Straight Fall	new training datasets (1278 raw datasets were collected)	Not available	window size of 20 frames equivalent to 0.5 sec	Training (50%), testing (50%) with data augmentation	Mean absolute percentage error (MAPE)	Maximum epochs: 125 Mini-batch size: 64 Weight initializer function: Glorot Solver: Adam Dropout rate: 0.2 Initial learning rate: $1x10^{-2}$ Gradient threshold: 2 Gradient threshold method: Global-l2norm L2 Regularization: $1x10^{-5}$	the data set was divided in half for training and testing but other datasets were created using n-fold (3) data augmentation approaches the 4-fold with All 3 augmentation gave the lowest error.
Hussain F et al., 2019	Young people, 23(-)(12 female - 11 male). Elderly users, 15(-)(7 female - 8 male)	two accelerometers (only one used in this study) (MMA8451Q and ADXL345) and one ITG3200 gyroscope	/	Feature extraction and K-Nearest Neighbors (KNN), Random Forest (RF), and Support Vector Machine (SVM) used for classification	Waist	19 ADLs (no fall class) and 15 fall actions (fall class)	SisFall dataset. (15 Falls, 19 ADLs) Fs=200Hz	publicly available	10 seconds using a non-overlapping window	10-fold cross-validation	Accuracy, Sensitivity, Specificity, Precision, F1-score	SVM with cubic kernel function and one-vs-one classification approach KNN classifier based on Euclidean distance with K=1 RF classifier with iterations set to 500	the study focus is on fall detection and fall recognition. Three classifiers were used (K-Nearest Neighbors (KNN), Random Forest (RF), and Support Vector Machine (SVM)). Butterworth filter was used. 4 statistical methods for Feature extraction used for fall-detection
shahzad A et al., 2019	20(28.45±2.72 years) (17 males - 3 females) (weight: 66.15±10.83 kg, and height: 170.7±7.68 cm)	accelerometer sensor embedded in Smart Phone	Smart Phone (LG G3 with a 3000-mAh battery ,Quad-core 2.7 GHz Krait 450 CPU, Adreno 420 GPU, 3-GBRAM and Android OS v4.4.2)	threshold-based method using acceleration vector magnitude (AVM) and multiple kernel learning support vector machine (MKL-SVM)	Waist (belt/ pouch) and thigh (pant pocket)	10 Falls (forward, backward, lateral, syncope, or vertical downfall with different body dynamics), 9 ADLs	209 simulated falls and 137 fall-like ADL events (waist), 175 simulated falls and 196 fall-like ADL events (thigh)	Not available	3sec window centered at the impact peak)	5-fold cross-validation	Accuracy, Sensitivity, Specificity	C=400 for waist and C=250 for thigh locations	15 Features extracted from AVM signal to classify Falls from ADL using MKL-SVM to improve the system sensitivity. The MKL-SVM was compared with 4 machine learning algorithms and it overperformed them. Battery estimated lifetime 86 h (24.32-h usage plus 62-h expected residual usage time)
Luna-Perejón F et al., 2019	Young people, 23(-)(12female - 11 male) Elderly people, 15(-)(7 female - 8 male)	TG3200 gyroscope, two accelerometers (models ADXL345 and MMA8451Q) Fs=200Hz	STM32L476RG (80Mhz, 1Mbyte Falsh, 128 Kbyte sRam) STM32F411RE (80 Mhz, 512 Kbyte Flash, 128 Kbyte sRam)	Two models were used and compared LSTM and GRU	Waist	19 ADLs (no fall class) and 15 fall actions (fall class)	SisFall	publicly available	265 samples window (i.e, 1.28 sec)	Train (80%), test (20%) with 10-fold cross validation	Accuracy, Sensitivity, Specificity, Precision, F1-score	LSTM: Learning rate = 0.0005, batch size = 32, RNN drop =0 GRU: Learning rate = 0.0005, batch size = 32, RNN drop =0	the original dataset Fs = 200Hz and it was downsampled to 25Hz, the best two models were (one layer of LSTM and the other model is Two layers GRU)
Ahmed M et al., 2017	140 people	Shimmer3 IMU evaluation kit (tri accelerometer, trigyrometer, temperature, pressure, and magnetometer sensors)	/	KNN, SVM and neural network	/	Fall, FallR, Sit, Walk, Stand	SMotion	Not available	5 sec	NN: Train (70%), validation (15%), Testing (15%) KNN and SVM: k-fold cross validation	Accuracy	KNN classifier based on Euclidean distance with K=3, SVM with kernel function set to Gaussian and multiclass method as one-vs-one, NN Levenberg-Marquardt optimization (Feedforward network)	KNN, SVM and neural network were assessed but KNN has the highest accuracy above them
Hsieh C et al., 2017	/	tri-axial (x,y,z) accelerometer (6g), Fs=128Hz	/	Threshold then SVM	Waist	7 ADLs, 7 Falls	collected dataset	Not available	4 sec (1.5sec before critical point and 2.5 sec after)	5-fold cross-validation	Accuracy, Sensitivity, Specificity, Precision	multicalss method set to one-versus-one and linear kernel function.	/

Table 3.3: Comparison between performance matrices results among included studies

Study, yr	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-score (%)**	**Sensitivity (%)**	**Specificity (%)**	**Testing Time (s)**	**average MAPE (%)**	**average correlation coefficient r**	**Sensor position**	**Other**
Lee Y et al., 2023	93.26	97.57	96.06	96.74	/	/	/	/	/	Chest	Young group
	82.44	84.46	86.82	85.37	/	/	/	/	/	Wrist	
	92.25	94.5	93.31	93.87	/	/	/	/	/	Necklace	
	93.53	95.52	95.78	95.61	/	/	/	/	/	Chest	Older group
Marques J et al., 2023	98.025	/	/	/	98.249	97.842	/	/	/	Waist	3NN
Al-qaness M et al., 2022	average CA =99.98, CA = 100	100	100	100	/	/	/	/	/	Waist	Fall
		100	100	100	/	/	/	/	/	Waist	no Fall
Ruiz J et al., 2022	93.51	/	/	/	92.04	95.45	/	/	/	Waist	/
Mankodiya H et al., 2022	92.54	/	/	/	74.8	98.3	/	/	/	Ankle, Chest, Right pocket, Waist, Wrist	/
Musci M et al., 2021	95.29	/	/	/	93.73	96.83	/	/	/	Waist	Validation set
	93.52	/	/	/	90.43	96.6	/	/	/	Waist	Testing set
Casilari E et al., 2020	99.4	/	/	/	98.8	99.7	/	/	/	Waist	ACC
	98.1	/	/	/	96.4	99.2	/	/	/	Waist	ACC+Gyr
Liu L et al., 2020	99.17	/	/	/	94.09	99.94	1.05	/	/	Waist	ACC+Gyr
Kim T et al., 2020	/	/	/	/	/	/	/	6.69±0.33	0.93	Waist	4-fold All augmentation methods
Hussain F et al., 2019	99.78	100	/	99.72	99.44	100	/	/	/	Waist	SVM
	99.56	99.55	/	99.44	99.33	99.7	/	/	/		Random Forest
	99.8	100	/	99.75	98.5	100	/	/	/		KNN
shahzad A et al., 2019	97.81	/	/	/	99.52	95.19	/	/	/	Waist	MKL-SVM
	91.7	/	/	/	95.83	88.01	/	/	/	thigh	
Luna-Perejón F et al., 2019	96.3	69.5	/	72.6	88.2	96.4	/	/	/	Waist	LSTM
	96.7	68.1	/	73	87.5	96.8	/	/	/		GRU
Ahmed M et al., 2017	96	/	/	/	/	/	/	/	/	/	KNN(age)
	93	/	/	/	/	/	/	/	/		KNN(Weight)
Hsieh C et al., 2017	99.33	99.05	/	/	99.79	98.74	/	/	/	Waist	knowledge based algorithm

Chapter 4

Materials and Methods

4.1 Introduction

Elderly individuals face an increased risk of falling as a result of natural age-related changes, and these falls pose significant medical dangers along with substantial healthcare and societal expenses. Nevertheless, there is a deficiency in automated fall detection systems tailored to the needs of older adults. Thus our work aimed to develop a neural network that is small enough to run on an embedded controller, and to improve the model to work with different datasets. moreover, the effect of adding a pressure sensor signal to the dataset was assessed.

This section will cover the equipment employed for data recording and decision-making, delve into the dataset utilized for model training, and ultimately discuss the suggested model along with the training approaches employed.

4.2 Hardware

4.2.1 Microcontroller

The "STM32U575xx" microcontroller Figure 4.1 was used, which belongs to an ultra-low-power microcontroller family known as the STM32U5 series. The microcontroller is based on the high-performance Arm Cortex-M33 32-bit RISC core and can operate at frequencies of up to 160 MHz but in our application we used an external 8 MHz crystal for power saving [48].

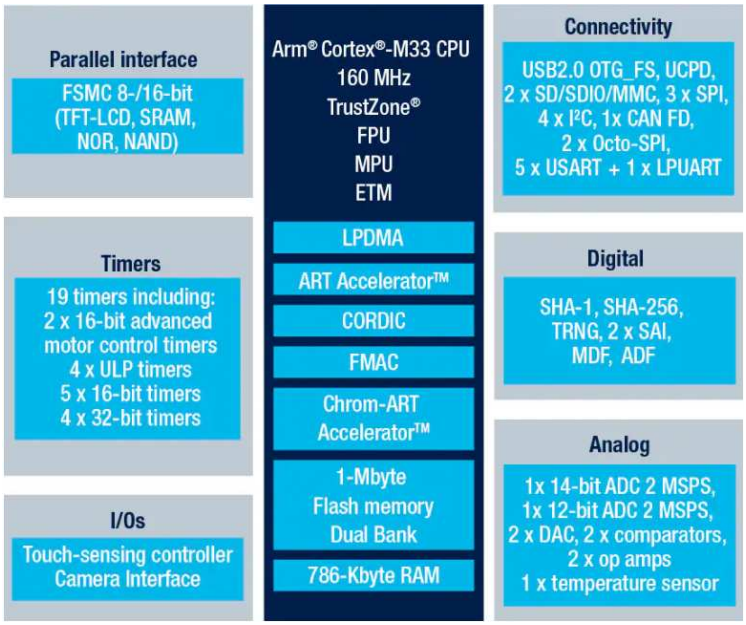

Figure 4.1: STM32U575xx circuit diagram

Main Features

The key features of the STM32U575xx microcontroller include:

- Single-precision FPU (floating-point unit) supporting Arm single-precision data-processing instructions and data types.
- Full set of DSP (digital signal processing) instructions.
- Memory Protection Unit (MPU) for enhanced application security.
- High-speed memories: Up to 2 MB flash memory and 786 KB SRAM.
- External memory interfaces: FSMC, Octo-SPI flash memory interfaces.

Security Features

The microcontroller offers robust security features:

- Trusted-Based Security Architecture (TBSA) compliance.
- Secure boot, secure data storage, and secure firmware update capabilities.
- Secure firmware installation during production.
- Readout protection, write protection and secure/hide protection areas.
- DPA resistance, HASH hardware accelerator, true random number generator.

Peripherals and Communication Interfaces

The microcontroller provides a comprehensive set of peripherals and communication interfaces, including:

- ADCs: 14-bit (2.5 Msps) and 12-bit (2.5 Msps).
- Comparators, operational amplifiers, DAC channels.
- Timers: General-purpose timers, PWM timers for motor control, low-power timers.
- Communication interfaces: I2C, SPI, USART, UART, SAI, USB OTG, FDCAN, etc.
- Digital signal processing: MDF (multifunction digital filter), ADF (audio digital filter), mathematical accelerators.

Power Management and Temperature Ranges

The microcontroller is designed for power efficiency and offers various power-saving modes. It operates in temperature ranges from -40 to +85°C (+105°C junction) and -40 to +125°C (+130°C junction) from a 1.71 to 3.6 V power supply (supplied with 3.3 V from a Li-ion battery 3.7 V and 11520 mAh).

Package Options

The STM32U575xx microcontroller is available in eight package options, ranging from 48 to 169 pins.

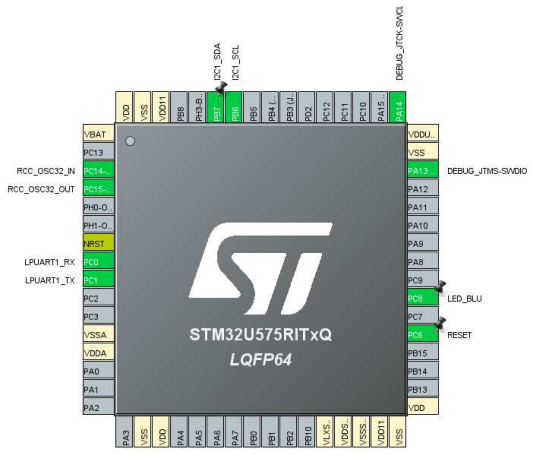

Figure 4.2: Pinout configuration of the STM microcontroller used in our project.

4.2.2 Inertial module

The inertial module (LSM6DSOX) employed in our project is a crucial component for motion tracking and fall detection featuring a 3D digital accelerometer and a 3D digital gyroscope. Below, the features and parameters used in our project's implementation of the LSM6DSOX inertial module are discussed [49].

Inertial module Features

- **Power Consumption**: 0.55 mA in combo high-performance mode
- **Always-On Experience**: Low power consumption for both accelerometer and gyroscope
- **Smart FIFO**: Up to 9 kbyte data storage with compression
- **Android Compliant**: Meets Android OS requirements
- **Full Scale Range**:
  - $\pm 2/\pm 4/\pm 8/\pm 16\,\mathrm{g}$ for linear acceleration
  - $\pm 125/\pm 250/\pm 500/\pm 1000/\pm 2000\,\mathrm{dps}$ for angular rate
- **Supply Voltage**:
  - Analog supply voltage: 1.71 V to 3.6 V
  - Independent IO supply: 1.62 V
- **Compact Footprint**: $2.5\,\mathrm{mm} \times 3\,\mathrm{mm} \times 0.83\,\mathrm{mm}$
- **Serial Interfaces**:
  - SPI / I²C & MIPI I3CSM for main processor data synchronization
  - Auxiliary SPI for OIS data output
- **Advanced Features**:
  - Pedometer, step detector, and step counter
  - Significant Motion Detection and Tilt detection
  - Standard interrupts: free-fall, wakeup, 6D/4D orientation, click and double-click
  - Programmable finite state machine for accelerometer, gyroscope, and external sensors
  - Machine Learning Core
  - S4S data synchronization
- **Embedded Temperature Sensor**
- **ECOPACK®, RoHS and "Green" Compliant**

Inertial module configuration

The accelerometer with 16 g and gyroscope with 2000 DPS resolution and 25 HZ sampling frequency were used.

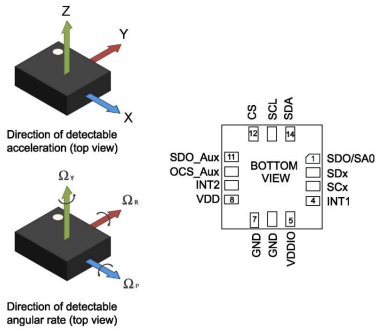

Figure 4.3: Pin connection of LSM6DSOX.

4.2.3 Pressure sensor

A pressure sensor (LPS25HB) was also used to monitor changes in the subject's height to study how this signal affected the fall detection algorithm and determine whether it could improve the model's accuracy in detecting falls.

Pressure sensor features

- 260 to 1260 hPa absolute pressure range
- High-resolution mode: 0.01 hPa RMS
- Low power consumption
  - Low-resolution mode: 4 μA
  - Low current & noise mode with FIFO: 4.5 μA
- High overpressure capability: 20x full scale
- Embedded temperature compensation
- 24-bit pressure data output
- ODR from 1 Hz to 25 Hz
- SPI and I2C interfaces
- Embedded FIFO
- Interrupt functions:
  - Data Ready
  - FIFO flags

  - Pressure thresholds

- Supply voltage: 1.7 to 3.6 V
- High shock survivability: 10,000 g
- ECOPACK® lead-free compliant. [50]

Pressure sensor configuration

The Pressure sensor with 16 samples moving average and 25 Hz sampling frequency was used.

Figure 4.4: LPS25HB sensor package.

4.2.4 Battery

A 3.7 V 1150 mA Li-ion battery was used to power the circuit.

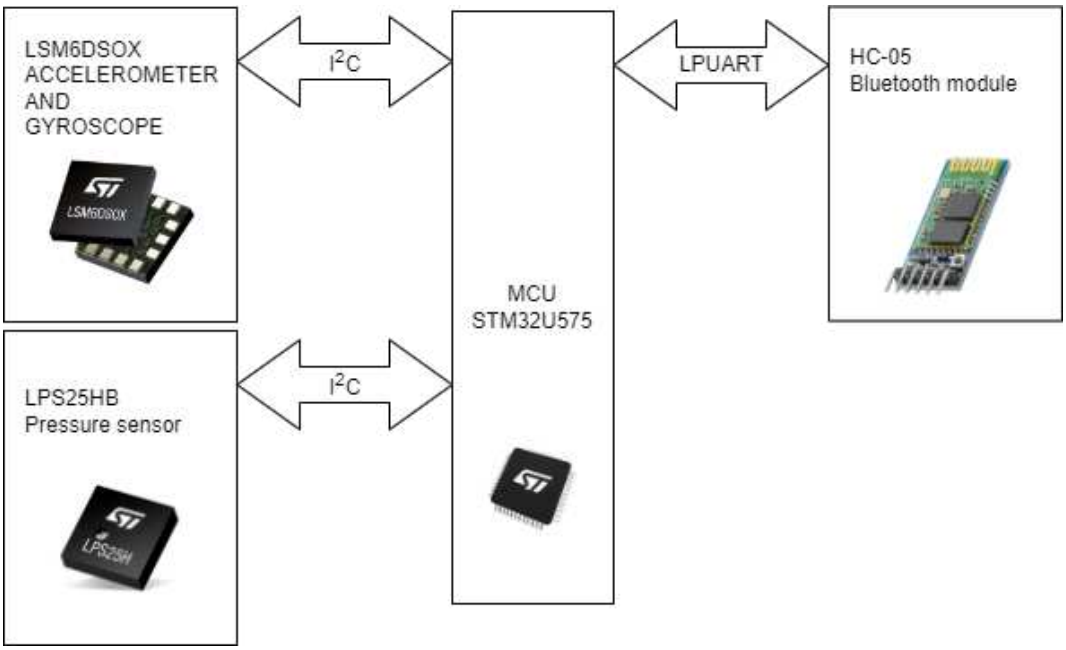

Figure 4.5: Block diagram of the wireless sensor node.

4.3 Datasets

Three distinct sets of data were employed for the model's training and testing. These include the publicly accessible SisFall dataset, a dataset we collected ourselves under controlled conditions, featuring additional sensor readings such as the pressure sensor that are absent in the SisFall dataset. Lastly, we merged our dataset with the SisFall dataset to enhance the model's accuracy and it is called the combined dataset. Further details about each of these datasets will be elaborated upon in the following paragraph.

4.3.1 SisFall dataset

We introduce the SisFall dataset, which contains recordings of falls and everyday activities (ADLs). The data was collected using a custom device equipped with two types of accelerometers and a gyroscope at a sampling frequency of 200 Hz downsampled to 25 Hz. The dataset includes a total of 19 different ADLs and 15 distinct types of falls in Table 4.2 and Table 4.1. The recordings were captured from 23 young adults, 14 older adults (aged over 62) performing 15 ADL types, and one 60-year-old participant who performed all ADLs and falls [51].

code	Activity	Trails	Duration
F01	Fall forward while walking caused by a slip	5	15s
F02	Fall backward while walking caused by a slip	5	15s
F03	Lateral fall while walking caused by a slip	5	15s
F04	Fall forward while walking caused by a trip	5	15s
F05	Fall forward while jogging caused by a trip	5	15s
F06	Vertical fall while walking caused by fainting	5	15s
F07	Fall while walking, with use of hands on a table to dampen fall, caused by fainting	5	15s
F08	Fall forward when trying to get up	5	15s
F09	Lateral fall when trying to get up	5	15s
F10	Fall forward when trying to sit down	5	15s
F11	Fall backward when trying to sit down	5	15s
F12	Lateral fall when trying to sit down	5	15s
F13	Fall forward while sitting, caused by fainting or falling asleep	5	15s
F14	Fall backward while sitting, caused by fainting or falling asleep	5	15s
F15	Lateral fall while sitting, caused by fainting or falling asleep	5	15s

Table 4.1: Types of falls included in the SisFall dataset [51].

code	Activity	Trails	Duration
D01	Walking slowly	1	100s
D02	Walking quickly	1	100s
D03	Jogging slowly	1	100s
D04	Jogging quickly	1	100s
D05	Walking upstairs and downstairs slowly	5	25s
D06	Walking upstairs and downstairs quickly	5	25s
D07	Slowly sit in a half-height chair, wait a moment, and up slowly	5	12s
D08	Quickly sit in a half-height chair, wait a moment, and up quickly	5	12s
D09	Slowly sit in a low height chair, wait a moment, and up slowly	5	12s
D10	Quickly sit in a low height chair, wait a moment, and up quickly	5	12s
D11	Sitting a moment, trying to get up, and collapse into a chair	5	12s
D12	Sitting a moment, lying slowly, wait a moment, and sit again	5	12s
D13	Sitting a moment, lying quickly, wait a moment, and sit again	5	12s
D14	Being on one's back change to the lateral position, wait a moment, and change to one's back	5	12s
D15	Standing, slowly bending at knees, and getting up	5	12s
D16	Standing, slowly bending without bending knees, and getting up	5	12s
D17	Standing, get into a car, remain seated, and get out of the car	5	25s
D18	Stumble while walking	5	12s
D19	Gently jump without falling (trying to reach a high object)	5	12s

Table 4.2: Types of ADLs (Activity of daily life) included in the SisFall dataset [51].

Participants

This dataset was created through the collaborative effort of 38 volunteers who were divided into two distinct categories: elderly individuals and young adults. The elderly group comprised 15 participants, with 8 being male and 7 being female. The young adults group consisted of 23 participants, with 11 males and 12 females. The age, weight, and height of each group are detailed in Table 4.3.

	Sex	Age	Height [m]	weight [kg]
Elderly	Female	62-75	1.50-1.69	50-72
	male	60-71	1.63-1.71	56-102
Adult	Female	19-30	1.49-1.69	42-63
	male	19-30	1.65-1.83	58-81

Table 4.3: Age, height, and weight of the participants in the SisFall dataset [51].

Experimental Set-Up

The dataset was captured using a self-developed embedded system made up of a Kinets MKL25Z128VLK4 microcontroller (NPX, Austin, Texas, USA), an Analog Devices ADXL345 accelerometer (configured for $\pm 16g$, 13 bits of analog to digital converter -ADC), a Freescale MMA8451Q accelerometer (for $\pm 8g$, 14 bits of ADC), an ITG3200 gyroscope (for $\pm 2000 degree/s$, 16 bits of ADC). The participants' waists were secured using the device Figure 4.6. In this area, a single accelerometer device can distinguish between various activities with great accuracy [51].

Figure 4.6: Device used for acquisition. The self-developed embedded device included two accelerometers and a gyroscope. It was fixed to the waist of the participants [51].

4.3.2 Self-collected dataset

We introduce a dataset containing instances of falls and everyday activities (ADLs) that we gathered using a custom-built device. This device is equipped with one accelerometer and gyroscope (LSM6dSOX), along with a pressure sensor (LPS25HB). We recorded the raw sensor data at a sampling rate of 25 Hz. Additionally, we implemented a 16-sample moving average feature within the pressure sensor to effectively mitigate noise in the data.

The dataset includes 18 ADLs and 16 fall types performed by 17 healthy young adults aged between 24 and 40 years and a weight between 50 and 90 kg. The data were collected in our laboratory after the candidates had been instructed about the protocol, and the aim of this study, and signed privacy questionnaires. The activities selected for this dataset are based on the SisFall dataset with an additional fall type namely Syncope (i.e. collapse or loss of consciousness [36]) as can be seen in Table 4.4 and Table 4.5. Integrating syncope detection into our fall detection system is crucial due to its high occurrence in emergency room visits, strong links to falls, and significant mortality rates in the elderly [52].

code	Activity	Trails	Duration
F01	Fall forward while walking caused by a slip	4	10s
F02	Fall backward while walking caused by a slip	4	10s
F03	Lateral fall while walking caused by a slip	4	10s
F04	Fall forward while walking caused by a trip	4	10s
F05	Fall forward while jogging caused by a trip	4	10s
F06	Vertical fall while walking caused by fainting	4	10s
F07	Fall while walking, with use of hands on a table to dampen fall, caused by fainting	4	10s
F08	Fall forward when trying to get up	4	10s
F09	Lateral fall when trying to get up	4	10s
F10	Fall forward when trying to sit down	4	10s
F11	Fall backward when trying to sit down	4	10s
F12	Lateral fall when trying to sit down	4	10s
F13	Fall forward while sitting, caused by fainting or falling asleep	4	10s
F14	Fall backward while sitting, caused by fainting or falling asleep	4	10s
F15	Lateral fall while sitting, caused by fainting or falling asleep	4	10s
F16	Syncope	4	10s

Table 4.4: Types of falls included in the self-collected dataset.

code	Activity	Trails	Duration
D01	Walking slowly	4	10s
D02	Walking quickly	4	10s
D03	Jogging slowly	4	10s
D04	Jogging quickly	4	10s
D05	Walking upstairs and downstairs slowly	4	10s
D06	Walking upstairs and downstairs quickly	4	10s
D07	Slowly sit in a half-height chair, wait a moment, and up slowly	4	10s
D08	Quickly sit in a half-height chair, wait a moment, and up quickly	4	10s
D09	Slowly sit in a low height chair, wait a moment, and up slowly	4	10s
D10	Quickly sit in a low height chair, wait a moment, and up quickly	4	10s
D11	Sitting a moment, trying to get up, and collapse into a chair	4	10s
D12	Sitting a moment, lying slowly, wait a moment, and sit again	4	10s
D13	Sitting a moment, lying quickly, wait a moment, and sit again	4	10s
D14	Being on one's back change to the lateral position, wait a moment, and change to one's back	4	10s
D15	Standing, slowly bending at knees, and getting up	4	10s
D16	Standing, slowly bending without bending knees, and getting up	4	10s
D17	Stumble while walking	4	10s
D18	Gently jump without falling (trying to reach a high object) 5 jumps	4	10s

Table 4.5: Types of ADLs (Activity of daily life) included in the self-collected dataset.

Participants

This dataset was collected from 17 volunteers, with 14 being males and 3 females. The age, weight, and height of each subject are detailed in Table 4.6.

SubjectID	Sex	Age	Height [cm]	weight [kg]
s01	male	27	169	50
s02	male	36	178	64
s03	male	27	175	75
s04	female	34	172	68
s05	male	31	174	65
s06	female	24	168	65
s07	female	26	170	65
s08	male	27	171	62
s09	male	30	171	64
s10	male	27	183	90
s11	male	30	176	70
s12	male	26	158	57
s13	male	29	178	74
s14	male	25	178	63.5
s15	male	24	183	75
s16	male	40	176	80
s17	male	24	177	90
Average	-	28.6	173.9	69.3

Table 4.6: Age, height, and weight of the participants in the self-collected dataset.

Experimental Set-Up

The dataset was captured using a self-developed embedded system made up of a microcontroller, and 3D accelerometer ($\pm 16g, 0.488mg/LSB, ODR26Hz$), and a gyroscope ($\pm 2000dps, 70mdps/LSB, ODR26Hz$)(LSM6dSOX), finally a pressure sensor (LPS25HB) ($ODR25Hz, AVGT = 16, AVGP = 32$, 16-sample moving average). The participants wore the device in their right pocket without securing it. 2 trials were done with the z-axis of the sensor in the direction of movement and 2 trials were done with the z-axis in the opposite direction. Figure 4.7 shows the device used to record the data and send it into a Matlab GUI Figure 4.8 to interpret and save the data as CSV files. STM32CubeIDE was used for the development of the code used on our STM32 microcontroller.

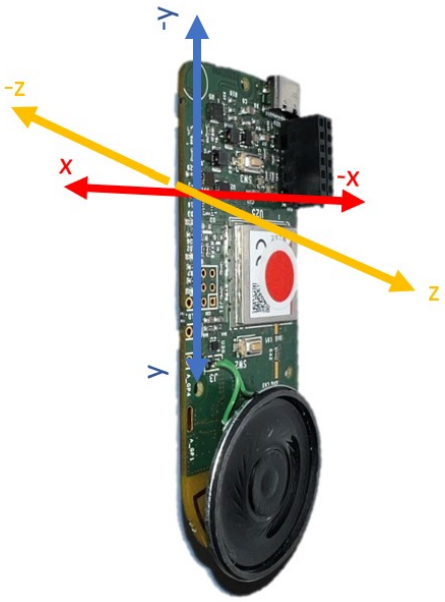

Figure 4.7: This image shows the device used to record the data and the axis of the device.

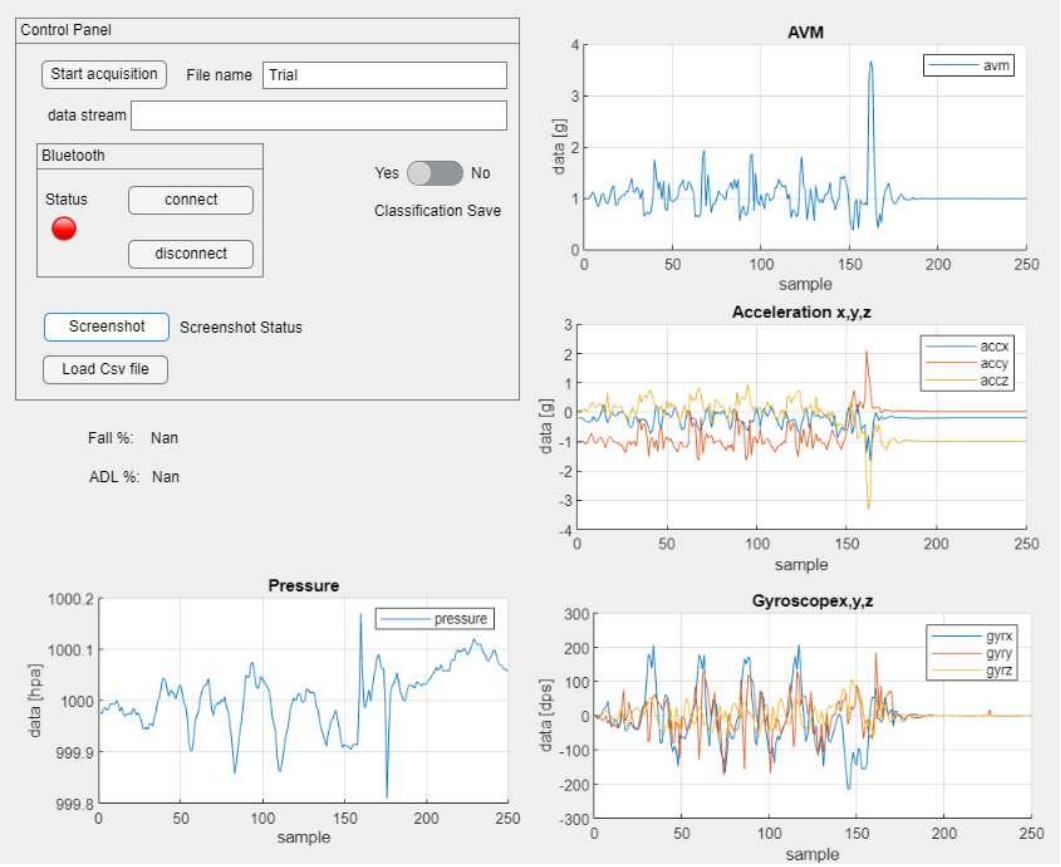

Figure 4.8: This image shows the Matlab GUI that has been built to record and interpret the data from the sensors.

Prototype testing setup

All tests and data collection were done in a controlled environment at Università Politecnica delle marche. The simulated Falls were performed on a (10 cm height - 200 cm length - 90 cm width) mattress for the subjects' safety.

Figure 4.9: Figure that shows an example of a fall forward while walking caused by a slip.

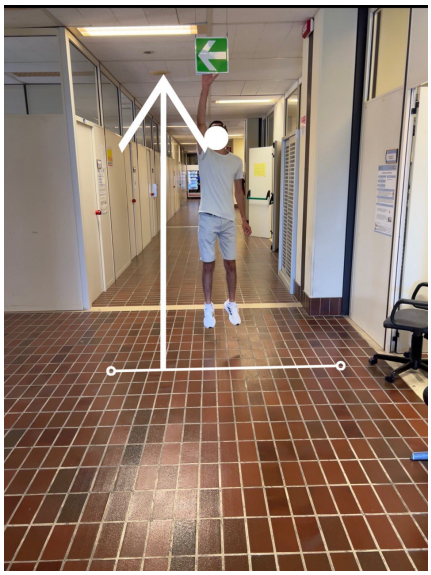

Figure 4.10: Figure that shows an example of gently jumping 5 times (try to reach something high).

Figure 4.11: Figure that shows an example of walking slowly.

Figure 4.12: Figure that shows an example of going up and down stairs slowly (5 steps).

4.4 Fall detection algorithm

4.4.1 Introduction

We will dive into the data collection, window preparation, and classification process. The Edge Impulse platform was used to develop the model due to its simplicity and ease of use and the ability to create lightweight deep-learning models that can run on various types of micro-controllers such as the STM microcontroller used in this project.

4.4.2 Data preparation

As previously mentioned, two datasets were used to train and test the model and its ability to discriminate between ADL and FALL classes, and the combination of the two datasets namely the combined dataset. It is an important aspect to control the window width and sampling frequency of our collected data that we will feed to the model to match the data used in training the model to maximize the accuracy of the model predictions.

Data preparation of SisFall dataset

The SisFall dataset contains 3D accelerometer data and 3D gyroscope data and for improving the model and generating a new signal that is orientation independent, the AVM (Acceleration Vector Magnitude) was calculated for each trial in the SisFall dataset as can be seen in Figure 4.13 and Figure 4.14.

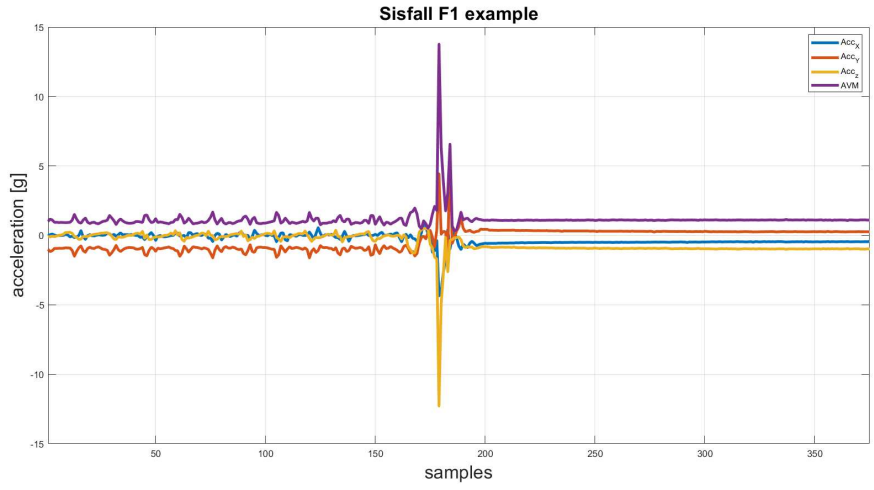

Figure 4.13: Fall sample from Sisfall dataset showing acceleration and AVM signals.

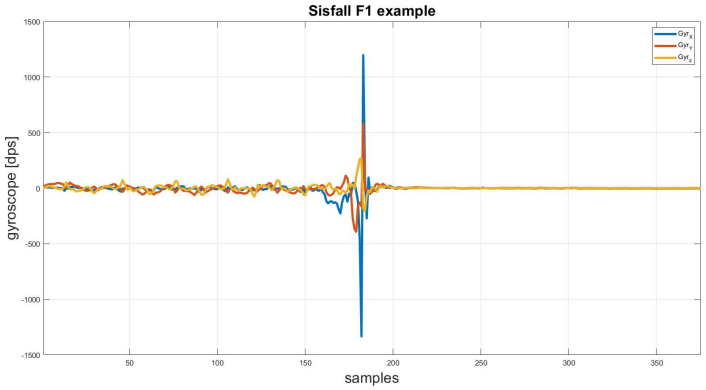

Figure 4.14: Fall sample from Sisfall dataset showing gyroscope signal.

Since the SisFall dataset contains different time windows for each task as can be seen in Table 4.2 and Table 4.1 which is sampled at 200 Hz, firstly SisFall dataset was downsampled to 25Hz to match our sensor configuration, and 12-sec window size was set for all tasks, we removed any signal that has a length less than 5 seconds since it doesn't contain important information (i.e. if a signal is 25 sec long, the signal is divided to 3 windows 12-12-1 seconds and the 1 second left over signal is removed).

After that set of features was calculated for each signal window that will be explained and listed later in the Spectral Analysis paragraph. These features will be used to train our model.

Data preparation of self-collected dataset

The same goes for the self-collected dataset, acceleration, gyroscope, AVM, and pressure signal were collected using our self-developed sensor node with a sampling frequency of 25 Hz and a window width of 10 seconds which is sufficient to capture the fall event. An example of the signals collected from our sensor node and will be used to be fed to the model can be seen in Figure 4.15, Figure 4.16, and Figure 4.17.

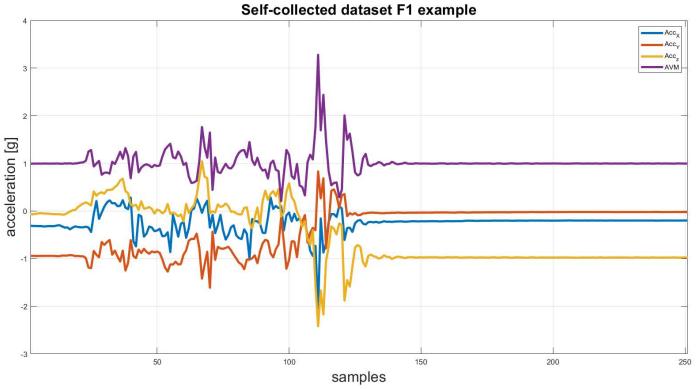

Figure 4.15: Fall sample from self-collected dataset showing accelerometer signal.

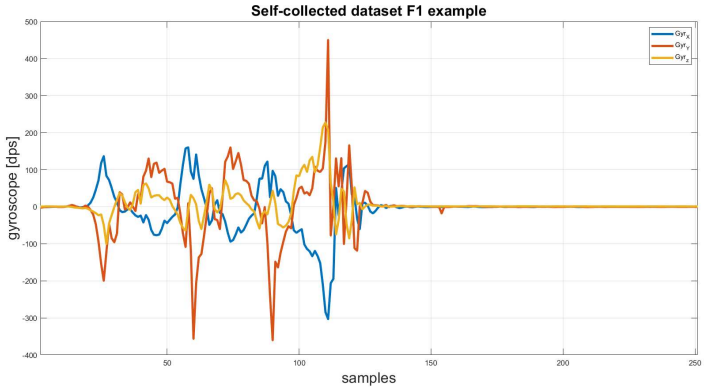

Figure 4.16: Fall sample from self-collected dataset showing gyroscope signal.

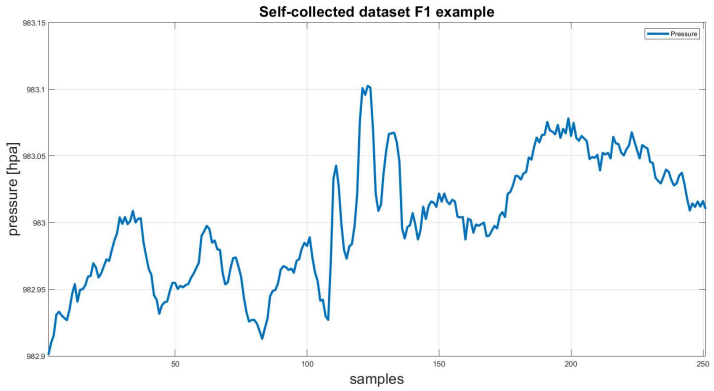

Figure 4.17: Fall sample from self-collected dataset showing pressure signal.

4.4.3 Model

Data acquisition

Data were divided into training set (80%) and testing set (20%). The training dataset was further divided into a (20%) validation set. We have made sure that both labels are included in the training and testing sets. The total length of the data was different for each of the 3 datasets used to train the model.

- SisFall dataset: 20h 12m 25s
- Self-collected dataset: 6h 26m 0s
- combined dataset: 26h 38m 35s

Impulse design

As previously mentioned Edge-Impulse was used for creating our deep learning model. For the input, a "Time series data" block was used with a 10,000 ms window size and a 2000 ms window increase and Fs = 25 Hz. If the data window is longer than 10,000 ms a sliding window with a 2000 ms increase will be applied to cover all the data windows. If it's less than 10,000 ms zero padding will be applied.

The time series data will be fed into a spectral analysis block to calculate features from them to be fed into the model as input, this block will be explained later in detail.

A classifier block will take the features generated by the spectral analysis block as input. The output of the classifier is either "ADL" or "Fall" class.

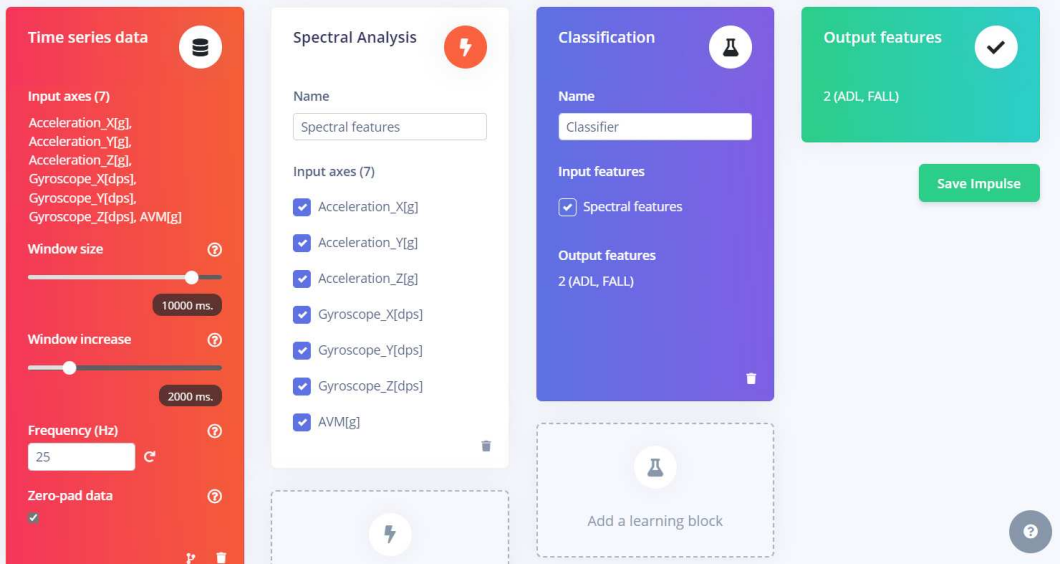

Figure 4.18: This figure shows the Impulse design window.

Spectral Analysis

In our classification task, the raw accelerometer data collected from sensors captures intricate movement patterns that signify either falls or routine activities. The data encompasses information in both the time and frequency domains. By applying spectral analysis to this dataset, our objective is to extract frequency-related features that can aid in distinguishing between falls and Activities of Daily Living (ADLs). This approach is particularly pertinent due to the potential distinct frequency characteristics of certain fall-associated movement patterns compared to typical ADLs.

Our analysis is conducted individually on each axis. We utilize Fast Fourier transformation, specifically employing 32 FFT length, for the spectral analysis.

Through this spectral analysis, we derive a set of features, for the Accx, Accy, Accz, Gyrx, Gyry, Gyrz, and AVM axes, and for our self-collected dataset, we also

include the pressure signal. Ultimately, we calculate a total of:

- 294 features for SisFall dataset (Since it has Acc(x,y,z), Gyr(x,y,z), AVM)
- 336 features for self-collected dataset (Since it has Acc(x,y,z), Gyr(x,y,z), AVM, and Pressure)
- 294 features for self-collected dataset without pressure (Since it has Acc(x,y,z), Gyr(x,y,z), AVM)
- 294 features for the combined dataset (Since it has Acc(x,y,z), Gyr(x,y,z), AVM)

Which are then utilized as input for our model. The feature selected for our task was inspired by literature [42, 18] and we built upon them.

Num	**Features**	**Num**	**Features**
1	Spectral Skewness	22	Spectral Power 4.3 - 5.08 Hz
2	Spectral Power 0.04 - 0.12 Hz	23	Spectral Power 0.51 - 0.59 Hz
3	Spectral Power 2.73 - 3.52 Hz	24	Spectral Power 1.17 - 1.95 Hz
4	Spectral Power 0.74 - 0.82 Hz	25	Spectral Power 0.27 - 0.35 Hz
5	Spectral Power 0.82 - 0.9 Hz	26	Spectral Skewness LF
6	Spectral Kurtosis LF	27	Skewness
7	Spectral Power 0.39 - 1.17 Hz	28	Spectral Power 0.9 - 0.98 Hz
8	Skewness LF	29	Spectral Power 0.43 - 0.51 Hz
9	Spectral Power 3.52 - 4.3 Hz	30	Spectral Kurtosis
10	Spectral Power 0.98 - 1.05 Hz	31	Kurtosis
11	Spectral Power 0.59 - 0.66 Hz	32	RMS
12	Spectral Power 1.95 - 2.73 Hz	33	Spectral Power 5.08 - 5.86 Hz
13	Spectral Power 0.2 - 0.27 Hz	34	Spectral Power 7.42 - 8.2 Hz
14	RMS LF	35	Spectral Power 8.2 - 8.98 Hz
15	Spectral Kurtosis LF	36	Spectral Power 5.86 - 6.64 Hz
16	Spectral Power 0.35 - 0.43 Hz	37	Spectral Power 12.11 - 12.89 Hz
17	Spectral Power 1.13 - 1.21 Hz	38	Spectral Power 10.55 - 11.33 Hz
18	Spectral Power 1.21 - 1.29 Hz	39	Spectral Power 6.64 - 7.42 Hz
19	Spectral Power 1.95 - 2.73 Hz	40	Spectral Power 8.98 - 9.77 Hz
20	Spectral Power 1.05 - 1.13 Hz	41	Spectral Power 0.66 - 0.74 Hz
21	Spectral Power 0.12 - 0.2 Hz	42	Spectral Power 9.77 - 10.55 Hz

Table 4.7: Feature Table

Classifier

For the model architecture, we used the Feed-Forward network which has a small memory footprint and achieved very good results for our task. Training Settings:

- Number of Training Cycles: 200
- Learning Rate: 0.0005
- Validation Set Size: 20%

The architecture of the neural network model used in this study is as shown in Figure 4.19 for combined dataset and sisfall dataset models, while the model input is slightly different for the self-collected dataset since it has an additional pressure signal as in Figure 4.20.

The model architecture is a Sequential neural network designed for classification. It comprises Dense layers with ReLU activation and dropout layers to prevent overfitting. The final layer is a softmax activation layer for multi-class classification. The training uses categorical cross-entropy loss and the Adam optimizer. This architecture is flexible and modular, incorporating the l1 regularization technique and dropout layers to enhance robustness.

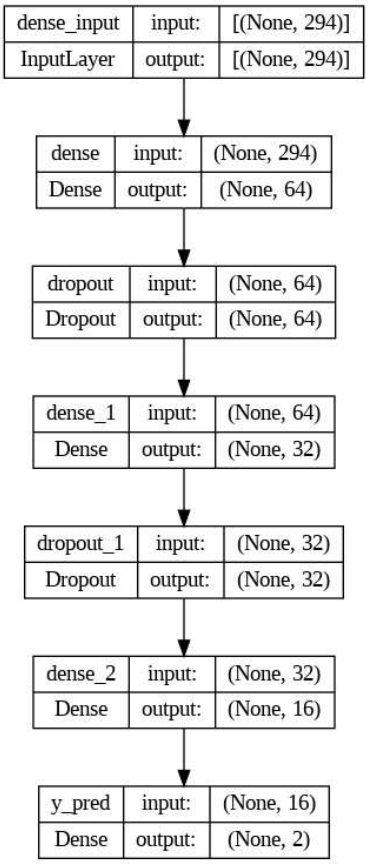

Figure 4.19: Model summary for sisFall and combined datasets.

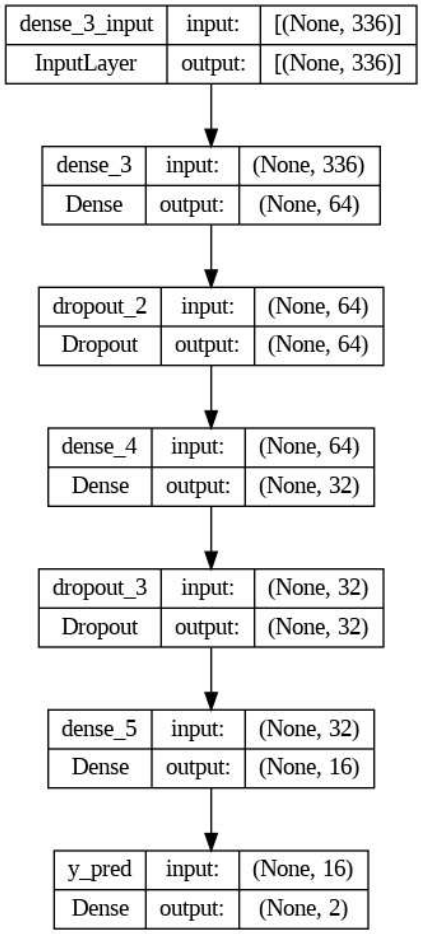

Figure 4.20: model summary for the self-collected dataset.

These hyperparameters and network architecture were employed to train and evaluate the performance of the classifier in distinguishing between the specified classes. A flow chart can be found in Figure 4.21 that describes the whole process from data collection until fall alert generation.

In Figure 4.21, the flow chart for the fall detection system is illustrated. Initially, sensor data is continuously collected. This data collection persists until the Accelerometer Vector Magnitude (AVM) signal surpasses a predefined threshold of 2.5 g. At this point, the probability of a fall occurrence is identified. From the moment the signal index exceeds the threshold, data collection continues until we collect 250 samples in total. Notably, the code is designed to ensure the fall event is centered within the recorded window. This approach aims to enhance model predictions by aligning the window with the training data. Subsequently, the recorded window undergoes feature extraction. These extracted features serve as inputs for the model, which generates probabilities for both falls and activities of daily living (ADLs). If the falling probability surpasses the ADL probability, a fall alarm is triggered.

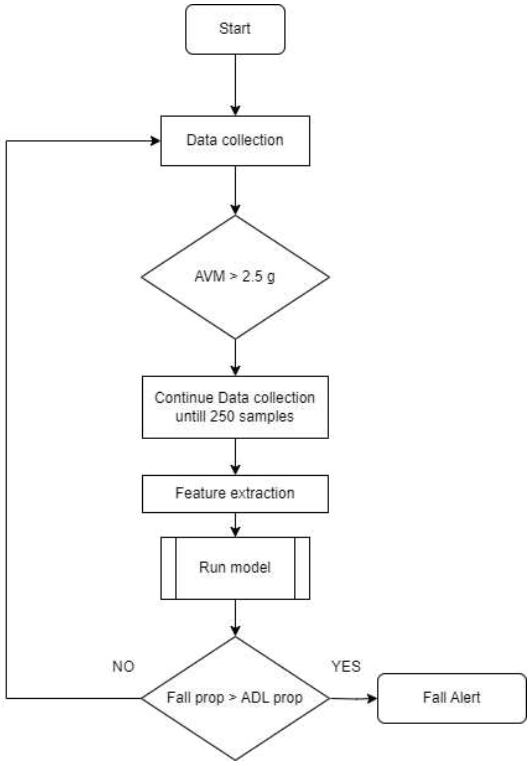

Figure 4.21: Flow chart of the proposed fall detection algorithm.

4.5 Conclusion

In this chapter, we discussed the hardware utilized for data collection and model execution. Subsequently, we delved into the datasets employed for training and evaluating our model. Lastly, we explored the model's architecture and the environment for conducting testing.

Chapter 5

Results and discussion

5.1 Results

5.1.1 Introduction

In this chapter, we will present the findings from the training and testing of our proposed model on various datasets, as well as the evaluation metrics.

5.1.2 Results of training and testing on SisFall dataset

The number of training data windows was 4843 with a length of 6 - 12 sec and 1211 testing data windows 10 - 12 sec. The total length of the data used for training and testing was 20h 12m 25sec with 80%/20% split, the training set was further split into validation set 20%. 294 features were extracted which are listed in Table 4.7.

Training accuracy was 99.7% with a loss of 0.02 on the validation set, results of testing are reported in Table 5.1. The inference time was 25 ms, with a peak RAM usage of 2.8 Kbyte, and Flash usage of 94.6 Kbyte.

5.1.3 Results of training on SisFall dataset and testing on self-collected dataset

The number of training data windows was 6054 with a length of 6 - 12 sec and 2316 testing data windows 10 sec. The total length of the data used for training and testing was 26h 38m 25sec with 80%/20% split, the training set was further split into validation set 20%. 294 features were extracted which are listed in Table 4.7.

Training accuracy was 99.8% with a loss of 0.01 on the validation set, results of testing are reported in Table 5.1. The inference time was 25 ms, with a peak RAM usage of 2.8 Kbyte, and Flash usage of 94.6 Kbyte.

5.1.4 Results of training and testing on self-collected dataset

In this dataset, we have in addition to the pressure signal, a new type of fall namely Syncope, and it was included in the training data. The number of training data windows was 1859 with a length of 10 sec and 457 testing data windows 10 sec. The total length of the data used for training and testing was 6h 26m with 80%/20%

split, the training set was further split into validation set 20%. 336 features were extracted which are listed in Table 4.7.

Training accuracy was 99.5% with a loss of 0.02 on the validation set, results of testing are reported in Table 5.1. The inference time was 25 ms, with a peak RAM usage of 3 Kbyte, and Flash usage of 105.1 Kbyte.

5.1.5 Results of training and testing on Combined dataset

As mentioned previously, in this dataset we combined our self-collected dataset (without the pressure signal) with the original SisFall dataset.

The number of training data windows was 6693 with a length of 5 - 12 sec and 1679 testing data windows 7 - 12 sec. The total length of the data used for training and testing was 26h 38m 35sec with 80%/20% split, the training set was further split into validation set 20%. 294 features were extracted which are listed in Table 4.7.

Training accuracy was 99.9% with a loss of 0.01 on the validation set, results of testing are reported in Table 5.1. The inference time was 25 ms, with a peak RAM usage of 2.8 Kbyte, and Flash usage of 94.6 Kbyte.

Training data	Testing data	Testing Acc(%)	TP(%)	TN(%)	FP(%)	FN(%)	F1_ADL	F1_FALL
SisFall	SisFall	99.3	99.3	99.3	0.7	0.7	0.99	0.99
SisFall	self-collected (without pressure)	94.86	97.3	99.9	2.3	6.7	0.95	0.95
self-collected (with pressure)	self-collected (with pressure)	99.12	99.1	99.2	0.9	0.4	0.99	0.99
self-collected (without pressure)	self-collected (without pressure)	98.05	98.1	98.0	1.9	2.0	0.98	0.98
Combined dataset	Combined dataset	99.38	98.5	99.8	1.3	0.2	1	0.99

Table 5.1: Results table (red row is the best dataset)

	ADL	FALL	UNCERTAIN
ADL	99.8%	0.2%	0%
FALL	1.3%	98.5%	0.2%
F1 SCORE	1.00	0.99	

Figure 5.1: Confusion matrix of "Combined dataset" testing.

5.1.6 Conclusion

Testing various datasets, each with its unique characteristics. Here are the key findings and conclusions drawn from our analysis:

1. **SisFall Dataset:** When training and testing on the SisFall dataset, our model achieved high accuracy, with a testing accuracy of 99.3%. This dataset provided an excellent baseline for evaluating fall detection capabilities.

2. **Self-Collected Dataset:** When we tested our model which was trained with the SisFall dataset on our self-collected dataset, excluding the pressure signal, it still demonstrated good performance with a slightly lower testing accuracy of 94.86%. Moreover, it showed slightly higher false negative/positive rates, indicating some room for improvement.

3. **Self-Collected Dataset with Pressure Signal:** The inclusion of the pressure signal in the self-collected dataset resulted in slightly higher accuracy and lower false positive/negative rate when compared with the self-collected dataset which was trained without the use of the pressure data, achieving a testing accuracy of 99.12% with the dataset which included the pressure signal.

4. **Combined Dataset:** The most promising results were obtained when combining our self-collected dataset (without pressure) with the original SisFall dataset. This combined dataset achieved the highest testing accuracy of 99.38% and exhibited the lowest false negative rate, making it the most effective dataset for fall detection.

5.2 Discussion

Our FFNN architecture stands in contrast to the recurrent architectures employed in the literature. Musci et al. (2021) [39] utilized Long Short-Term Memory Networks (LSTM), a type of recurrent neural network (RNN) well-suited for sequential data due to their ability to capture long-term dependencies. Shahzad et al. (2019) [36] leveraged Multiple Kernel Learning-Support Vector Machine (MKL-SVM), combining the power of kernel methods with the robustness of SVM. Luna-Perejón et al. (2019) [16] opted for Gated Recurrent Units (GRU), a variant of LSTM known for its computational efficiency.

In contrast, our FFNN model adopts a simple and efficient architecture without recurrent connections. The absence of recurrent loops renders our model computationally lightweight, enabling faster execution on resource-constrained microcontrollers. This architectural simplicity ensures faster inference times, making it well-suited for real-time applications where prompt detection of falls is crucial.

Our feature extraction strategy uses spectral analysis, capturing complex frequency patterns within accelerometer and gyroscope data, tHis approach inherently caters to the diverse and subtle movement patterns associated with falls, enhancing the model's discriminatory power. In contrast, Musci et al. (2021) [39] and Luna-Perejón et al. (2019) [16] primarily relied on raw sensor data, leveraging the temporal patterns learned by LSTM and GRU networks. Shahzad et al. (2019) [36] used a combination of threshold-based methods and pattern recognition techniques, focusing on accelerometer data.

A pivotal distinction lies in the implementation platform. Musci et al. (2021) [39], Shahzad et al. (2019) [36], and Luna-Perejón et al. (2019) [16] adapted their models for execution on microcontrollers, aligning with the embedded system paradigm. However, their models use recurrent architectures, which inherently demand more computational resources due to the recurrent connections.

In contrast, our FFNN model does not use recurrent loops, which minimizes computational power usage. This inherent efficiency facilitates swift execution

on microcontrollers, ensuring real-time fall detection. The absence of recurrent connections reduces memory requirements, enabling seamless integration into resource-constrained environments.

When examining performance metrics, our FFNN model emerges as the best model. It achieves a high testing accuracy of 99.38%. Musci et al. (2021) [39] reported an accuracy of 93.52%, Shahzad et al. (2019) [36] achieved 97.81%, and Luna-Perejón et al. (2019) [16] reached 96.7%. Our model's superiority in accuracy underscores its efficacy in precisely distinguishing falls from non-fall activities.

Moreover, our model showcases high sensitivity (99.79%) and specificity (98.62%), outperforming existing approaches. The high sensitivity emphasizes our model's proficiency in correctly identifying fall instances, crucial for timely intervention. Simultaneously, the impressive specificity underscores its accuracy in discerning non-fall scenarios, mitigating false alarms, and finally, comparing our model inference time of 25 ms with Luna-Perejón et al. (2019) [16] which is 34 ms.

Regarding the dataset selection Musci et al. (2021) [39] used the SisFall dataset which was further annotated, while Luna-Perejón et al. (2019) [16] collected their own dataset, for our study we used the SisFall dataset in combination with the Self-collected dataset with further adjustments where we calculated AVM for the whole dataset which improved model performance and also the target age group which can use our fall detection system as can be seen in Tabels 5.2 and 5.1.

In our study, there are notable limitations that deserve attention. Firstly, the process of protocol recording proved to be time-consuming, and participants often exhibited signs of fatigue afterward. This fatigue could potentially have influenced their falls, leading to discrepancies in the data collected. Such exhaustion might impact the authenticity of the falls simulated during the recording sessions.

Furthermore, our dataset labeled 'self-collected with pressure' was restricted to a younger demographic, lacking representation from elderly subjects. As predicting falls among the elderly is a crucial aspect of our research, the absence of this demographic in the 'self-collected with pressure' dataset posed a significant limitation. To address this limitation, we made the decision to exclude the pressure signal and integrate the available data with the extensive SisFall dataset, which includes a substantial number of falls from elderly participants. This integration resulted in the creation of our 'combined dataset' which can be used with a larger scale of age groups (i.e. elderly and young).

These limitations underscore the challenges faced during the data collection process and highlight the necessity for careful consideration when interpreting our results. While our approach mitigated some of these limitations, it is important for future studies to explore methods for minimizing participant fatigue during protocol recording and to ensure a more comprehensive representation of age groups in the datasets, particularly when studying fall detection among the elderly population.

Study	Model	MCU Implementation	Testing Acc(%)	TP(%)	TN(%)	FP(%)	FN(%)	F1_ADL	F1_FALL	Sensitivity (%)	Specificity (%)	Precision (%)	Inference time(ms)
Our approach	FFNN	Yes	99.38	98.5	99.8	1.3	0.2	1	0.99	99.79	98.62	98.69	25
Musci M et al.(2021)	LSTM	Yes	93.52	/	/	/	/	/	/	90.43	96.6	/	/
shahzad A et al.(2019)	MKL-SVM	Yes (smart phone)	97.81	/	/	/	/	/	/	99.52	95.19	/	/
Luna-Perejón F et al.(2019)	GRU	Yes	96.7	/	/	/	/	/	/	87.5	96.8	68.1	34

Table 5.2: comparison between our best model and the literature.

Chapter 6

Conclusion

In conclusion, our study delved into the evaluation of a fall detection system, emphasizing the significance of a simple yet effective model architecture suitable for embedded controllers. We carefully designed our model using dense layers with a small memory footprint, ensuring its feasibility for real-world applications. The choice of a sampling frequency at 25 Hz allowed for low-power data collection while maintaining accurate fall detection capabilities.

Our research stands out in the landscape of fall detection studies for several reasons. Comparing our findings with the existing literature, our model, despite its simplicity, demonstrated exceptional accuracy and robustness with fast inference time. In the realm of fall detection, where complex approaches are prevalent, our results underscore the effectiveness of streamlined architectures, especially in resource-constrained environments.

One of the unique contributions of our work lies in the expansion and integration of datasets. By adding a new type of fall to the self-collected dataset, Syncope, calculating AVM for all datasets, and integrating the self-collected dataset with the SisFall dataset which increased the target age groups that can use our fall detection system, our combined dataset emerged as a powerhouse for training our fall detection model. The incorporation of diverse fall types, including the introduction of the new fall Syncope, in our combined dataset expanded the scope of fall patterns studied. This diversity played a pivotal role in enhancing the adaptability and versatility of our trained model. By training on this enriched dataset, our model became more proficient in recognizing a wide range of fall scenarios, making it highly versatile and capable of handling different real-world fall situations effectively.

Furthermore, our study shed light on the pivotal role of pressure data in enhancing fall detection accuracy. The inclusion of pressure signals led to a small improvement in the model accuracy, highlighting the importance of multifaceted data inputs. This finding resonates with recent research emphasizing the significance of incorporating diverse sensor modalities for comprehensive fall detection systems [42].

Looking forward, our work paves the way for future endeavors in the realm of fall detection. The limitations of existing datasets, especially concerning the involvement of elderly subjects, pose challenges that need to be addressed collectively by the research community. Exploring a wider array of sampling frequencies and time

windows remains a promising avenue for fine-tuning fall detection models.

The deployment of our proposed device holds the potential to significantly enhance the safety and well-being of individuals at risk of falling, thereby making a substantial impact on public health.

Bibliography

[1] DA SINGH. Human balance and posture control during standing and walking. *Gait & Posture*, 3(4):193–214, 1995.

[2] Anuradha Singh, Saeed Ur Rehman, Sira Yongchareon, and Peter Han Joo Chong. Sensor technologies for fall detection systems: A review. *IEEE Sensors Journal*, 20(13):6889–6919, 2020.

[3] Bijan Najafi, David Horn, Stefan Marclay, Ryan T Crews, Samuel Wu, and James S Wrobel. Assessing postural control and postural control strategy in diabetes patients using innovative and wearable technology. *Journal of Diabetes Science and Technology*, 4(4):780–791, 7 2010.

[4] Robert J. Peterka. Chapter 2 - sensory integration for human balance control. In Brian L. Day and Stephen R. Lord, editors, *Balance, Gait, and Falls*, volume 159 of *Handbook of Clinical Neurology*, pages 27–42. Elsevier, 2018.

[5] Francesco Carini, Margherita Mazzola, Chiara Fici, Salvatore Palmeri, Massimo Messina, Provvidenza Damiani, and Giovanni Tomasello. Posture and posturology, anatomical and physiological profiles: overview and current state of art. *Acta Biomed*, 88(1):11–16, Apr 2017.

[6] J Massion. Postural control system. *Current Opinion in Neurobiology*, 4(6):877–887, Dec 1994. PubMed PMID: 7888772.

[7] Kaoru Takakusaki. Functional neuroanatomy for posture and gait control. *Journal of Movement Disorders*, 10(1):1–17, Jan 2017. PubMed PMID: 28122432, PubMed Central PMCID: PMC5288669.

[8] Marcos Duarte and Sandra M S F Freitas. Revision of posturography based on force plate for balance evaluation. *Revista brasileira de fisioterapia (Sao Carlos (Sao Paulo, Brazil))*, 14(3):183–92, May-Jun 2010.

[9] Jia-Li Sung, Lan-Yuen Guo, Chin-Hsuan Liu, Posen Lee, Chen-Wen Yen, and Lih-Jiun Liaw. Assessing postural stability using coupling strengths between center of pressure and its ground reaction force components. *Applied Sciences*, 10(22), 2020.

[10] Johanna Vielemeyer, Cristina Sole, Manuela Galli, Matteo Zago, Roy Müller, and Claudia Condoluci. A study on the intersection of ground reaction forces

during overground walking in down syndrome: Effects of the pathology and left–right asymmetry. *Symmetry*, 15(2), 2023.

[11] David A. Winter, Aftab E. Patla, Francois Prince, Milad Ishac, and Krystyna Gielo-Perczak. Stiffness control of balance in quiet standing. *Journal of Neurophysiology*, 80(3):1211–1221, 1998. PMID: 9744933.

[12] D. A. Winter, A. E. Patla, and J. S. Frank. Assessment of balance control in humans. *Medical progress through technology*, 16(1-2):31–51, May 1990.

[13] Xinguo Yu. Approaches and principles of fall detection for elderly and patient. In *HealthCom 2008 - 10th International Conference on e-health Networking, Applications and Services*, pages 42–47, 2008.

[14] Centers for Disease Control, National Center for Injury Prevention Prevention, and Control. Keep on your feet—preventing older adult falls. *Keep on Your Feet—Preventing Older Adult Falls*, 2023.

[15] Harsh Vasoya, Hetvi Bhattasana, and Raj Gaurav Mishra. A review of elderly fall detection systems using artificial intelligence. In *2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 541–546, 2023.

[16] Francisco Luna-Perejón, Manuel Jesús Domínguez-Morales, and Antón Civit-Balcells. Wearable fall detector using recurrent neural networks. *Sensors (Switzerland)*, 19, 11 2019.

[17] Han Wen Guo, Yi Ta Hsieh, Yu Shun Huang, Jen Chien Chien, Koichi Haraikawa, and Jiann Shing Shieh. A threshold-based algorithm of fall detection using a wearable device with tri-axial accelerometer and gyroscope. In *2015 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*, pages 54–57, 2015.

[18] Chia Yeh Hsieh, Kai Chun Liu, Chih Ning Huang, Woei Chyn Chu, and Chia Tai Chan. Novel hierarchical fall detection algorithm using a multiphase fall model. *Sensors (Switzerland)*, 17, 2 2017.

[19] Paola Pierleoni, Alberto Belli, Lorenzo Palma, Marco Pellegrini, Luca Pernini, and Simone Valenti. A high reliability wearable device for elderly fall detection. *IEEE Sensors Journal*, 15(8):4544–4553, 2015.

[20] Raquel Leirós-Rodríguez, Jose L García-Soidán, and Vicente Romo-Pérez. Analyzing the use of accelerometers as a method of early diagnosis of alterations in balance in elderly people: A systematic review. *Sensors*, 19(18):3883, 2019.

[21] Zakriya Mohammed, Ibrahim (Abe) Elfadel, and Mahmoud Rasras. Monolithic multi degree of freedom (mdof) capacitive mems accelerometers. *Micromachines*, 9:602, 11 2018.

[22] Vittorio M. N. Passaro, Antonello Cuccovillo, Lorenzo Vaiani, Martino De Carlo, and Carlo Edoardo Campanella. Gyroscope technology and applications: A review in the industrial perspective. *Sensors (Basel, Switzerland)*, 17(10):2284, 2017.

[23] `https://www.mouser.com/blog/set-yourself-straight-imus` Accessed on 2023-08-12.

[24] Imtiaz Hossain Sarker, Md-Hafizul Islam Furhad, and Ripon Nowrozy. Ai-driven cybersecurity: An overview, security intelligence modeling and research directions. *SN Computer Science*, 2, 2021.

[25] Iqbal H. Sarker. Machine learning: Algorithms, real-world applications and research directions. *SN Computer Science*, 2(3):160, 2021.

[26] Mohssen Mohammed, Muhammad Khan, and Eihab Bashier. *Machine Learning: Algorithms and Applications*. CRC group, 07 2016.

[27] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[28] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015.

[29] R.E. Uhrig. Introduction to artificial neural networks. In *Proceedings of IECON '95 - 21st Annual Conference on IEEE Industrial Electronics*, volume 1, pages 33–37 vol.1, 1995.

[30] Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. Activation functions: Comparison of trends in practice and research for deep learning, 2018.

[31] Frank Emmert-Streib, Zhen Yang, Han Feng, Shailesh Tripathi, and Matthias Dehmer. An introductory review of deep learning for prediction models with big data. *Frontiers in Artificial Intelligence*, 3:4, 02 2020.

[32] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[33] `https://www.educative.io/answers/keras-dense-layer` Accessed on 2023-08-12.

[34] `https://it.mathworks.com/campaigns/offers/deep-learning-with-matlab.html` Accessed on 2023-08-27.

[35] Lingmei Ren and Yanjun Peng. Research of fall detection and fall prevention technologies: A systematic review. *IEEE Access*, 7:77702–77722, 2019.

[36] Ahsan Shahzad and Kiseon Kim. Falldroid: An automated smart-phone-based fall detection system using multiple kernel learning. *IEEE Transactions on Industrial Informatics*, 15:35–44, 1 2019.

[37] Mohammed A.A. Al-qaness, Ahmed M. Helmi, Abdelghani Dahou, and Mohamed Abd Elaziz. The applications of metaheuristics for human activity recognition and fall detection using wearable sensors: A comprehensive analysis. *Biosensors*, 12, 10 2022.

[38] Harsh Mankodiya, Dhairya Jadav, Rajesh Gupta, Sudeep Tanwar, Abdullah Alharbi, Amr Tolba, Bogdan Constantin Neagu, and Maria Simona Raboaca. Xai-fall: Explainable ai for fall detection on wearable devices using sequence models and xai techniques. *Mathematics*, 10, 6 2022.

[39] Mirto Musci, Daniele De Martini, Nicola Blago, Tullio Facchinetti, and Marco Piastra. Online fall detection using recurrent neural networks on smart wearable devices. *IEEE Transactions on Emerging Topics in Computing*, 9:1276–1289, 2021.

[40] Leyuan Liu, Yibin Hou, Jian He, Jonathan Lungu, and Ruihai Dong. An energy-efficient fall detection method based on fd-dnn for elderly people. *Sensors (Switzerland)*, 20:1–16, 8 2020.

[41] Tae Hyong Kim, Ahnryul Choi, Hyun Mu Heo, Hyunggun Kim, and Joung Hwan Mun. Acceleration magnitude at impact following loss of balance can be estimated using deep learning model. *Sensors (Switzerland)*, 20:1–17, 11 2020.

[42] Faisal Hussain, Fawad Hussain, Muhammad Ehatisham-Ul-Haq, and Muhammad Awais Azam. Activity-aware fall detection and recognition based on wearable sensors. *IEEE Sensors Journal*, 19:4528–4536, 6 2019.

[43] Yongkuk Lee, Suresh Pokharel, Asra Al Muslim, Dukka B. KC, Kyoung Hag Lee, and Woon-Hong Yeo. Experimental study: Deep learning-based fall monitoring among older adults with skin-wearable electronics. *Sensors*, 23:3983, 4 2023.

[44] Eduardo Casilari, Moisés Álvarez Marco, and Francisco García-Lagos. A study of the use of gyroscope measurements in wearable fall detection systems. *Symmetry*, 12, 4 2020.

[45] João Marques and Plinio Moreno. Online fall detection using wrist devices. *Sensors*, 23, 2 2023.

[46] Jesús Fernández Bermejo Ruiz, Javier Dorado Chaparro, Maria José Santofimia Romero, Félix Jesús Villanueva Molina, Xavier Del Toro García, Cristina Bolaños

Peño, Henry Llumiguano Solano, Sara Colantonio, Francisco Flórez-Revuelta, and Juan Carlos López. Bedtime monitoring for fall detection and prevention in older adults. *International Journal of Environmental Research and Public Health*, 19, 6 2022.

[47] Moiz Ahmed, Nadeem Mehmood, Adnan Nadeem, Amir Mehmood, and Kashif Rizwan. Fall detection system for the elderly based on the classification of shimmer sensor prototype data. *Healthcare Informatics Research*, 23:147–158, 2017.

[48] `https://www.st.com/en/microcontrollers-microprocessors/stm32u575rg.html` Accessed on 2023-08-20.

[49] `https://www.st.com/en/mems-and-sensors/lsm6dsox.html` Accessed on 2023-08-20.

[50] `https://www.st.com/en/mems-and-sensors/lps25hb.html#overview` Accessed on 2023-09-08.

[51] Angela Sucerquia, José David López, and Jesús Francisco Vargas-Bonilla. Sisfall: A fall and movement dataset. *Sensors (Basel, Switzerland)*, 17(1):198, Jan 2017.

[52] Parag Goyal and Mathew S Maurer. Syncope in older adults. *J Geriatr Cardiol*, 13(5):380–6, 2016.