Afonso Manuel
Macedo Guimarães

BedFeeling - Sensing Technologies for Assistive Communication in Bed Scenarios

BedFeeling - Tecnologias de Sensorização para Apoio à Comunicação no Cenário da Cama

Afonso Manuel
Macedo Guimarães

BedFeeling - Tecnologias de sensorização para apoio à comunicação no cenário da cama

BedFeeling - Sensing Technologies for Assistive Communication in Bed Scenarios

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor Ilídio Castro Oliveira, Professor Auxiliar do Departamento Eletrónica Telecomunicações e Informática da Universidade de Aveiro, e do Doutor António Joaquim da Silva Teixeira, Professor Associado com Agregação do Departamento Eletrónica Telecomunicações e Informática da Universidade de Aveiro

o júri / the jury

presidente / president	**Professor Doutor Joaquim João Estrela Ribeiro Silvestre Madeira**, Professor Auxiliar da Universidade de Aveiro
vogais / examiners committee	**Professor Doutor Ilídio de Castro Oliveira** Professor Auxiliar da Universidade de Aveiro (Orientador)
	Professor Doutor Hugo José Sereno Lopes Ferreira Professor Auxiliar da Faculdade de Engenharia da Universidade do Porto

agradecimentos / acknowledgements

Gostaria de agradecer a todos que direta e indiretamente fizeram parte no processo de desenvolvimento desta dissertação.

Um especial obrigado ao Prof. Doutor Ilídio Castro Oliveira, que me deu a oportunidade de ingressar no projeto, que acreditou em mim e que me deu orientação ao longo de todos estes meses.

Um agradecimento ao Prof. Doutor António Teixeira, que me acompanhou com afinco em todo o processo de desenvolvimento desta dissertação, por todo o conhecimento, apoio e motivação transmitidos.

Queria aproveitar também para agradecer à Doutora Ana Rocha, por todo o apoio, conhecimento transmitidos na área e especialmente por toda a paciência demonstrada.

Gostaria de agradecer aos restantes elementos da equipa do IETTA percententes ao projeto do APH-Alarm que criaram um ambiente crítico e de inter-ajuda que foram importantes para o desenvolvimento do trabalho realizado.

Fica também um enorme agradecimento a todos os participantes das experiências realizadas, porque sem o seu apoio e tempo dispendido a obtenção de resultados não era possível.

Finalmente gostaria também de agradecer aos meus pais, irmã, namorada e amigos, especialmente ao Vasco, Rui e Tiago. Obrigado pela confiança e por todo o apoio nos momentos mais críticos.

Palavras-Chave

Afasia, Ambientes Inteligentes, Comunicação, Gestos, Sensores, Cenários da Cama, Aprendizagem Automática

Resumo

Pessoas com deficiências na fala, como a afasia, enfrentam geralmente dificuldades para recuperar e manter a sua independência enquanto vivem uma vida ativa. O cenário da cama assume grande relevância para estes indivíduos, devido às diferentes dificuldades que podem ocorrer e requerer assistência (e.g., dor súbita, dificuldade de movimentação de membros e, neste contexto, é importante avaliar soluções para essa população.

Esta dissertação tem como objetivo a conceptualização e criação de uma solução baseada em sensores que reconheça movimentos dinâmicos do braço, com o utilizador deitado na cama, de forma a apoiar a comunicação, proporcionando meios não só para o alarme de uma emergência, mas também para permitir uma comunicação bidirecional e simples entre um afásico e o seu cuidador.

A solução desenvolvida usa um único "smartwatch" para detectar um conjunto de gestos usando os valores do acelerómetro, giroscópio, e magnetómetro, fornecidos através do smartwatch a uma unidade de processamento colocada ao lado da cama. Esta unidade é responsável por receber, classificar e decidir se qualquer movimento realizado é um dos movimentos dinâmicos predefinidos. Se positivo, esta unidade envia um alerta ao cuidador por meio de uma aplicação móvel especialmente desenvolvida para o efeito e que permite enviar perguntas básicas, de resposta sim ou não, de volta ao afásico às quais ele pode responder usando, novamente, um dos movimentos dinâmicos suportados.

O sistema, que pode ser subdividido em três módulos, inclui uma aplicação para um smartwatch responsável por adquirir os dados e enviá-los para uma unidade de processamento, um pipeline implementado na unidade de processamento responsável por receber e classificar os dados, enviando o resultado para uma aplicação móvel na posse do cuidador, e uma aplicação móvel desenvolvido para o cuidador receber as notificações e permitir o envio de mensagens ao utilizador sob sua responsabilidade. A unidade de processamento também fornece feedback sonoro, permitindo ao afásico receber indicações do seu cuidador. Os resultados para reconhecimento de gestos são bastante positivos, quer para um cenário de "dependência de sujeito" ou "independência de sujeito" (i.e. "accuracy" e F1-score com média a rondar os 99% e 91% respetivamente), mostrando que uma solução generalizada pode ser alcançada, tornando as duas abordagens viáveis.

Embora o sistema tenha sido construído para uso por utilizadores afásicos, este não se limita unicamente aos mesmos, podendo ser generalizado para cenários que requeiram a ligação entre pessoas na cama aos seus cuidadores, quando o uso da voz não for viável ou prático.

Key Words Aphasia, Smart Environments, Communication, Gestures, Sensors, In-Bed Scenarios, Machine Learning

Abstract

People with certain speech impediments, such as aphasia, face challenges to keep independent and active lives. The bedroom scenario assumes a strong relevance for these individuals due to the different difficulties that can occur and require assistance while in bed. In this context, it is important to pursue assistive solutions for this population.

In this dissertation, we aim at creating a sensor-based solution that recognizes dynamic arm movements while in bed, to support communication, providing ways to raise alarms to some hazard conditions, and also enable bidirectional and simple communication between aphasics and their caregivers.

The solution developed uses a common smartwatch to collect movement data (using the built-in accelerometer, gyroscope, and magnetometer), which are forwarded to a bedside unit for processing. The bedside unit is responsible for receiving, classifying, and deciding if the movement performed at any time is one of the supported predefined dynamic arm movements. If positive, the bedside unit sends an alert to the caregiver using a specially developed mobile application that allows sending basic "yes" or "no" questions back to the aphasic, to which he may respond using the supported dynamic arm movements.

The system, which may be subdivided into three modules, includes a smartwatch app responsible for acquiring the data and sending them to a bed-side unit, a pipeline implemented on the bed-side unit responsible for receiving and classifying the data using previously trained machine learning models, and sending the result to a mobile app in the possession of the caregiver, and a mobile app developed for the caregiver to receive the notifications and allow sending messages to the user being cared for. The bedside unit also implements a speech output service that provides audio near the bed, allowing the aphasic to hear feedback from the caregiver. The gesture recognition results are encouraging, both for subject-dependent and subject-independent scenarios (i.e. mean accuracy and F1-score above 99% and 91% respectively), showing that a model generalization may be attained, making both approaches feasible.

Although the system was built for the use case of aphasics, it is not limited to those users and can be generalized for scenarios that require connecting people in bed to their caregivers, when the use of voice is not feasible or practical.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Aphasia is a disturbance of the comprehension and formulation of language caused by dysfunction in specific brain regions [1]. People with aphasia demonstrate challenges in all areas of communication, from reading and writing to comprehension and expressive language [2]. This condition makes it hard for this population to regain and keep their independence, influencing everyday living and working as a barrier to communicate potential problematic or difficult situations to family, friends, authorities, and health institutions. These individuals can greatly benefit from assistive communication solutions, namely those that can support communication in bed but also throughout the day.

The bed scenario assumes a strong relevance for these individuals, due to the different difficulties that can occur and require assistance while awake during the night or resting during the day. In this context, aspects relating to body movement, gestures, and pose can potentially play a role as alternative ways to communicate specific needs or difficulties, both implicitly and explicitly. In this regard, the proposal of solutions based on sensors that are minimally intrusive, comprehensive, reliable, and easily integrated into the bed environment of the target population, can bring a plethora of benefits assisting them with their needs and providing straightforward mechanisms to communicate needs.

Previous work established for aphasics and other senior populations was evaluated and, although there are some solutions to support monitoring of people while lying in bed, to the best of our knowledge, there is not one that tackles the communication aspect of the situation and so lies an opportunity to start developing solutions that can support aphasics and other groups with communication barriers in a bed context.

1.2 Context

The work developed for this dissertation was motivated by the research activities in the scope of the APH-ALARM project, Comprehensive safety solution for people with Aphasia (AAL/0006/2019)[1]. APH-ALARM is an international project, in the context of the AAL Programme, with the contributions of partners from Portugal, Hungary, and Austria, including the University of Aveiro, through its IEETA research unit.

[1] `http://www.aal-europe.eu/projects/aph-alarm/`

The purpose of the project is to create a comprehensive safety solution for older people (55+) with impairments such as aphasia, epilepsy, and/or side-paralysis after a stroke that can be used throughout the day. To allow support throughout the day (outside the bedroom), the solution will support manual alert systems using pictograms and automatic alerts using activity and gesture detection using a smartphone with accessibility options for people affected by communication impairments. To support this population while lying in bed, the solution will rely on sensors to detect movements that allow aphasics or others to communicate with, for instance, a caregiver or family member, without interacting directly with a device.

In this context, this dissertation aims at exploring the in-bed scenario, where the use of movement sensors will be assessed as a potential solution to support communication for people with aphasia or other language impediments.

1.3 Objectives

The fundamental goal of this dissertation is to provide a communication mechanism based on sensors for aphasics, while in bed. The system should be able to detect meaningful gestures from users laying in bed and forward those events to caregivers, for scenarios in which it is not possible or practical to use the voice or make selections in an application, to call for help or basic assistance.

This system should be easily integrated, non-intrusive, and capable of providing accurate and fast responses for communication attempts by this group of aphasics, generally, a more elderly group, offering to them a sense of independence and warmth while also giving their families and caregivers a sense of security.

To achieve this overall goal, a few more specific sub-goals will be pursued:

- Study the specific requirements of the Aphasic-related use cases, and the most relevant communication scenarios to support. The activities of the APH-ALARM project will provide valuable inputs.
- Define a controlled set of arm gestures that are appropriate for the in-bed situation. Unlike other smart home settings, the gestures must be easy to execute when the user is laying in bed and can include the mattress.
- Develop a gesture classification model, using machine learning techniques and data collected with the proposed experimental setup.
- Integrate the sensing and classification capabilities to enable simple communication with another person (*e.g.,* a caregiver).

The setup should be minimally intrusive without compromising the user's well-being, contributing to a sense of security, by connecting people being cared for with their caregivers.

1.4 Contributions

The main contribution of this dissertation is the development of a system based on classical sensors, to support communication for people suffering from aphasia after stroke. This system is based on the usage of a single wearable that collects data, and a processing unit that performs the gesture recognition feature as well as implementing the communication channels

with the caregiver via the Android caregiver application, which was also developed for this dissertation.

The gesture recognition feature developed in this dissertation recognizes five gestures that are simple to execute in the bed context and, in certain circumstances, employ the mattress as an intermediary. The models used for gesture recognition were generated by assessing a set of sensor data acquired from a variety of people using the setup and methods detailed later in this dissertation.

The final gesture recognition models were integrated into a functional system that allows users to interact with a caregiver who is attending to them. With the system deployed, when a meaningful gesture is performed, an alert is transmitted to the caregiver. The caregiver's application allows them to send not only confirmation of the receipt of communication attempts, but also simple yes/no questions, which originate voice feedback to the speakers present in the bedroom of the user, who may respond using the supported gestures.

Other contributions involve a paper describing the preliminary results obtained for gesture recognition using an early setup, which was published at the IEEE International Smart Cities Conference 2021, held online from September 7 to September 10, 2021 [3], contributions as co-author on a paper describing a setup with a gesture recognition module, submitted to PerCom 2022 (with notification on December 22, 2021), and contributions on a paper describing an approach to gesture recognition in a bed-scenario using radar, submitted and accepted to EAI MobiQuitous 2021.

1.5 Structure

This dissertation is divided into five chapters, excluding this one. **Chapter 2** provides some background and related work on the subjects studied in this dissertation, mainly on those related to aphasics and gesture recognition. **Chapter 3** presents the requirements of the system, as well as the definition of personas and scenarios related to the system, *i.e.,* hypothetical subjects and scenarios that entail the use of the system and people affected by it. **Chapter 4** presents the envisioned solution and all the processes involved in its development. The findings of the gesture recognition module implemented for the system are presented in **Chapter 5**, which is separated into two sections, each containing the results of two studies that led to the models used in the final solution. The steps taken during the development of the solution, the main conclusions, and possible future work are presented in **Chapter 6**.

Chapter 2

Background and Related Work

This section provides an overview of Aphasia as a condition that can impact people after a stroke, as well as the current state of the art in assistive technology specially developed for individuals with speech impediments, where aphasics are included.

Also presented are specifics on the concept of gesture recognition, a very relevant topic for the work developed throughout this dissertation, with an analysis of a large series of works in order to assess the current state of the art in the area of gesture recognition.

2.1 Aphasia

Aphasia is a reality predominantly unknown for the general population (approximately 84.5% of people never heard the term and only 8.8% can identify it as a language disorder [4]), and although having a lot of controversy in what concerns its definition, it is mostly defined as being the loss or impairment of language caused by brain damage (Benson & Ardila, 1996 [5]). This brain damage usually occurs due to a stroke and it is characterized as a sudden loss of speech [6], although some of the individuals may recover from it on the first few weeks from the stroke. The ones that do not recover will be permanently affected by language disorders that depend on the location and extent of the brain damage.

Aphasia limits the ability to communicate and interact with others in many degrees, and the majority of individuals suffering from it present language disorders that may affect listening, comprehension, expression, reading, and writing ([7, 6]).

As stated before, the location and extent of the brain damage influence the type of aphasia and therefore different symptoms affecting the individual. Despite the variability of lesion sites across persons with the same aphasia type, lesion patterns within a particular aphasia type are comparable enough to distinguish it from other aphasia types [8]. Table 2.1 lists the symptoms of the predominant types of aphasia that affect people after stroke.

Even in the most severe forms of aphasia, recovery is possible. While speech-language therapy is still the most common treatment for aphasia, the efficacy of traditional treatments has not been shown definitively. This has prompted efforts to combine information from many fields to develop more reasonable therapies and introduce new therapeutic techniques, such as intensive language therapy and pharmaceutical medicines [9].

Because of the different barriers that affect these individuals, they often have their jobs, relationships and day-to-day life damaged, leading to embarrassment and depression, and so, assistive technologies specially developed for these individuals and others affected by speech

Type of Aphasia	Speech Fluency	Object nomination	Understanding of Simple Orders	Word Repetition
Broca's Aphasia	Non-fluent	Disturbed	Kept	Disturbed
Wernicke	Fluent	Disturbed	Disturbed	Disturbed
Transcortical Sensory	Fluent	Disturbed	Disturbed	Kept
Conduction	Fluent	Disturbed	Kept	Disturbed
Anomic	Fluent	Disturbed	Kept	Kept
Global	Non-fluent	Disturbed	Disturbed	Disturbed

Table 2.1: Most common types of aphasia and their symptoms.

disabilities are important to improve their lives.

2.2 Assistive Technology for Aphasia

Establishing new methods to communicate is a crucial part of speech therapy. These new methods usually involve gestures, drawing, writing, or the use of assistive technologies. Any approach, strategy, technology, or device that complements, augments or substitutes speech to assist persons with limited speech abilities is referred to as Augmentative and Alternative Communication (AAC) [10, 11].

Most AAC technology is designed to assist people to express their necessities (e.g. "I'm in pain...") and usually consists of specially crafted devices or apps used to allow a speech-disabled person to communicate. According to the sensing strategy employed, these AAC technologies may be classified into five primary types [10]: imaging; mechanical and electromechanical; touch-activated; breath-activated; brain-computer interface. All of these approaches convert the communication attempts into readable signs, images, or voice outputs (using Text-to-speech technology) that can be interpreted by other users [12]. A more detailed description of each method is presented below.

1. **Imaging methods** use vision-based sensors (e.g., RGB, RGB-D, or infrared cameras) to enable eye or head tracking for eye gazing or head-pointing activation. Examples of commercially available eye gaze/tracking AAC systems are those provided by Tobii Dynavox [13], eyespeak™ [14], and IntelliGaze [15]. They can typically be used together with other input modalities (e.g., switch access, head tracking, touchscreen) [10].

2. **Mechanical and electromechanical methods** rely on mechanical keyboards or switches, which are used for direct or indirect selection access, respectively. They are often integrated with other devices (e.g., computers, tablets). Lingraphica is an example, which offers devices built for stroke or brain injury survivors and other users with communication impairments, providing practice tools for speech improvements and other tools to help improve the lives of those that deal with these problems daily [16].

3. **Touch-activated methods** rely on touchscreens or touch membrane keyboards, and are used for direct selection activation, where the user selects letters or icons correspond-

ing to words/expressions to write text or form sentences [10]. There are several commercially AAC touchscreen applications available for use together with a (non-)dedicated tablet or other smart devices, including Verbally (iOS only) [17], Proloquo2Go (iOS only) [18], SmallTalk (iOS only) [19], and Predictable™ [20].

These AAC methods, except for imaging, have the problem of not being suitable for all situations and times of the day. Although fine for day situations where the user is feeling well and has the devices in his possession, when considering a situation where the user is lying in bed, resting during the day or awake at night, it may not be easy to use systems involving a keyboard/switch and/or a device with a (touch)screen. That is where sensor-based solutions are recalled that may be used throughout the whole day, being able to track movements or other data from the users.

Breath-activated methods encode messages through modulation of the speed, amplitude, and phase of breathing signals, which can be detected using different types of sensors (e.g., microphones, pressure, or thermal sensors). These methods can be used, for instance, to turn into Morse Code and therefore encode to text [10].

Brain-computer interface (BCI) methods are frequently used to allow the control of external devices [10]. Non-invasive BCIs rely on external equipment to monitor a user's brain activity, such as EEG. [10]. However, despite the fact that EEG recording methods are non-invasive and inexpensive, they are nonetheless highly obtrusive since they require the installation of two or more sensors on the user's scalp.

Even those with limited motor abilities can employ the stated techniques: eye/head tracking, breath activation, and a brain-computer interface. Nevertheless, although they mostly use non-invasive methods, they can still be very intrusive for the user, due to the placement of sensors on the body (e.g., on the user's scalp for BCI) or the use of vision-based sensors that can compromise privacy (e.g., RGB or infrared cameras for eye/head tracking). Some also need significant training and calibration prior to usage [10].

When motor function is not a problem or only one side of the body is affected, arm/hand gestures and wearable sensors (such as a smartwatch, which many individuals already have and use daily) may be a good AAC option for in-bed use. The recognition of sign language has already been proposed for helping with communication [21]. However, the use of more simple gestures, such as dynamic arm movements, may be more adequate for people with speech impairments, who may just want a simple way to alert for a situation or to indicate something pre-defined that the caregiver/friend/family and the aphasic both understand.

In that regard, gesture recognition may be introduced and give an important opportunity to understand if a solution based on gestures may be used to ease the lives of those affected by speech impairments, where aphasics are included. This would allow the translation of simple hand/arm movements to, for instance, text messages, that may be sent to a designated caregiver, and therefore allow them to communicate more easily.

2.3 Gesture-Based Human-Computer Interaction

Human activity recognition (HAR) is a general area of research that focuses on recognizing a person's individual activity by analysing movement data, captured by sensor-based systems. These activity recognition systems can be integrated in a plethora of scenarios, from solutions that envision the recognition of walking patterns between different users [22], to solutions that foresee the recognition of user gestures.

Research in the specific area of gesture recognition has, as stated various applications [23, 24], including human-computer interaction (HCI) (e.g., control of home devices or a car) [25], human-robot collaboration [26], virtual/augmented reality [27], healthcare (e.g., system for healthcare muscle exercise) [28], and communication (e.g., sign language) [22]. The majority of contributions in the field of communication are concerned with the identification of sign language gestures [22, 21, 29].

Most action or gesture-based interaction systems are either implemented using computer vision methods or using wearable sensor data. Computer vision methods using different types of vision-based sensors, such as RGB, RGB-D/depth, infrared, and thermal cameras, have been widely used in research [24, 23], while lately radars have also been explored [25]. Although effective for the recognition of gestures, the use of computer vision methods is not cost efficient and is computationally expensive, being only viable in a controlled environment with a constant video monitoring solution in place, which is not attainable in every scenario, introducing compromises regarding privacy and, in most cases, are very sensitive to changes on the environment (e.g. light, position).

Sensors used for hand/arm gesture recognition include wearable sensors [23], such as accelerometers, gyroscopes, and surface electromyography (sEMG) sensors embedded in gloves or smartwatches. These sensors, meant to be worn by the user, have the benefit of not generating privacy issues over radar or vision-based sensors. Moreover, while wearables introduce the need for the user to remember to wear and recharge them, they may offer continuous monitoring independently of time and location, and do not require line of sight between the sensor and the user's hands/arms, unlike radars and cameras deployed in the environment [25], and therefore are less sensitive to the environment they are in.

Regarding hand or arm gesture recognition using wearables, several works aim at HCI [30, 31, 32, 33, 34, 35, 36, 37, 38, 39]. Many examples include uses for remote control of home appliances [40, 36], healthcare contexts [33], gaming [35], wheelchair control [41], robot interaction [42, 43, 39], or interaction with mobile phones by people with visual impairments [44].

Gesture-based systems that rely on the use of wearable sensors usually have a comparable approach, using similar gestures, sensors, and even recognition methods. Examples of some of this systems, described in [45, 44, 30], use a setup containing only a wearable with an accelerometer to detect hand gestures such as left, right, down, up, square and circles. However, more recent researches tend to use a combination of sensors, mostly accelerometers and gyroscopes [46, 47], to detect the same set of gestures or even more complex arm movements.

Tchuente et al. used smartwatches with an accelerometer and gyroscope to investigate the classification of aggressive and non-aggressive movements such as punching, shoving, slapping, shaking [46]. The goals were to find the best location for the wearable sensor, as well as the best combination of feature selector and classifier, to achieve high precision in the identification of motions such as punch, shove, slap, and shake, to name a few. It was found that having a single smartwatch worn on the dominant wrist and using a combination of RelieF for feature selection, and k-Nearest Neighbor for classification, held a very good accuracy of 99.6%, using a one second sliding window and a 96% window overlap.

Porzi et al. suggested a gesture recognition system for people with visual impairments that used a smartwatch with built-in accelerometers [44]. To assess their models, they used Support Vector Machines (SVM), Global Alignment Kernels, and a Dynamic Time Warping model. They were able to recognize eight complex movements, involving up, down, left, right, and square, among others, with an accuracy of 92.33% using a custom implementation

of Support Vector Machines (SVM) together with a global alignment kernel (GAK) .

Although the contributions referenced up until now used machine learning for gesture recognition with excellent results, others have used deep learning, where the motion signals are fed directly to the neural networks [47, 42]. A sensor-based hand gesture recognition system based on Feed-Forward and Convolutional Neural Networks using sensory data produced by accelerometers and gyroscopes was used, getting accuracy's up to 92.36% for a set of 11 gestures with over 3404 test gestures [47], while in a hand gesture recognition system, based only on 3D accelerometers was used with a Recurrent Neural Network getting accuracy's up to 99% for a 6 class gesture set [42]. Deep learning solutions have the disadvantage of requiring a large amount of data to be trained and tested, as well as demanding more computing power to do so; however, for more complex scenarios where machine learning fails to deliver good results, these solutions tend to demonstrate that progress can be made in the topic, with the growth of complex neural networks.

Considering preprocessing techniques, more specifically the choice of window size and window overlap, there is some investigation done tackling it. Banos et al. analyzed some of the most frequently used activity recognition techniques for a wide variety of window sizes and activities for activity recognition. According to the results, the interval 1–2 seconds provides the greatest balance of identification speed and accuracy [48]. Chikhaoui has also demonstrated that for the identification of aggressive and agitated behavior using accelerometers, a small window size of around one second is justified [49]. The choice of window overlap allows to obtain more training data but can also lead to over fitted models, Dehghani et al. has made a comparison of overlapping and non-overlapping sliding windows for human activity recognition and the results showed that there is no performance gain from the use of overlapping windows in conjunction with subject-independent cross validation [50]. This result has an high impact on resource usage in the feature selection and training process of models. Although these conclusions were taken, it is important to assess if in the scenario tackled in this dissertation the same is observable.

Tables 2.2 and 2.3 summarize the state of the art in activity and gesture recognition research, with a focus on the publications that are most relevant to this dissertation. Details on the goal, sensors used, features extracted, classifiers used, and classification results are presented.

Ref.	Year	Authors	Context	Sensor(s)	Position
[45]	2011	Tea Marasović et al.	System that uses the accelerometer, embedded in a mobile phone, to capture simple gestures, such as hand describing a circle.	Accelerometer	Standing
[44]	2013	Porzi et al.	Gesture-based user interaction module, based on global alignment kernels, for assisting people with visual impairments during daily life activities.	Accelerometer	Standing
[30]	2017	Kefer et al.	Effect of device placement on dynamic hand gesture recognition accuracy.	Accelerometer	Standing
[51]	2018	Zhu et al.	Action detection and segmentation algorithm that is used in a system that uses common smartwatches to infer gestures in real time.	Gyroscope, Accelerometer	Standing
[42]	2018	Carfi et al.	Architecture for online gesture recognition, based on a wearable triaxial accelerometer, a Recurrent Neural Network (RNN) probabilistic classifier and a procedure for continuous gesture detection	Accelerometer	Standing
[47]	2020	Chu et al.	Study on the use of Neural Network algorithms to accurately classify a sequence of hand gestures from the sensory data produced by accelerometers and gyroscopes.	Gyroscope, Accelerometer	Standing
[52]	2020	Siddiqui et al.	Study on hand gesture recognition with the use of acoustic signals together with an accelerometer and gyroscope at the human wrist.	Gyroscope, Accelerometer and Array of Microphones	Standing
[46]	2020	Tchuente et al.	Study on the detection of aggressive movements using smartwatch data. This also studied the placement of the sensor units, and investigated if one wrist-worn smartwatch was enough.	Gyroscope, Accelerometer	Standing

Table 2.2: State of art table concerning gesture recognition, with specifics on the context, sensors and position of the user

Ref.	Gestures supported	Features	Classifiers	Accuracy
[45]	Right, Down, Square, Circle, Triangle, L-Shape, N-Shape	**For each axis:** Average, Average Absolute Difference, Variance, Standard Deviation, Root Mean Square, Zero-Crossings, Signal-to-Noise Ratio. Duration, Correlation Coefficient, Acceleration, Average Resultant, Binned Distribution	kNN with PCA LDA	87.6% using LDA
[44]	Left, Right, Up, Down, Circle Clockwise, Circle Counterclockwise, Square, Arrow To The Right	Global Alignment Kernel	SVM, DTW	92.33% using a custom Global Alignment kernel together with SVM
[30]	Left, Right, Down, Up, Square, Circle Counter Clockwise, Circle Clockwise, Triangle	**For each axis:** Mean, Min, Max, Range, Variance, Magnitude, Peaks, Energy, Frequency Range, Gesture Duration	SVM, J48 decision tree, Naive Bayes, and Neural Networks.	90.73% with Leave-Subject-Out Cross-Validation
[51]	Wave four fingers right, point with index finger, clicking the index finger and the thumb finger, rubbing the thumb finger with other four fingers updown, fingers in the shape of calling .	Frequency Domain Features	Recurrent Neural Network (RNN)	96% using LSTM based Neural Network
[42]	Wrist twist, arm up and down, circle movement	The inertial signals are fed directly.	Recurrent Neural Network (RNN)	96.9% for offline testing
[47]	Left, Right, Up, Down, , Circle Clockwise, Triangle, Bolt Shape, S-Shape UP, S-Shape Down, Slight Curve	The inertial signals are fed directly.	PairNet, Residual PairNet, PairNet wih Inception, Residual PairNet wih Inception, CNN, LSTM, Bi-LSTM	92.36% with Pairnet
[52]	Static hand gestures (eg. hand lift, thumbs up, okay sign)	7873 features with some of the top being shannon entropy, burstiness, second-order moment, highlowmu statistic, interquartile range.	SVM (Gaussian kernel) with 10-fold cross-validation; LDA classifier with Monte-Carlo cross-validation	75%
[46]	Punch, shove, slap, shake, clap, wave, handshake, type on keyboard.	**For each axis:** Mean, Variance, Median, Range, Standard Deviation, Skewness, Kurtosis, Parwise Correlation Coeficient, Integral	k-Nearest Neighbors (kNN), Multilayer Perceptron Neural Network (MP), Support Vector Machine (SVM), Naïve Bayes, decision tree	99.6% using a combination of RelieF for feature extraction and kNN.

Table 2.3: State of art table concerning gesture recognition, with specifics on the gestures, features, classifiers and results attained.

2.4 Critical Analysis on Gestures for Communication

It is challenging to locate a contribution that addresses both the in-bed scenario discussed in this research as well as the communication needs of people with speech impairments like aphasia. Even though these issues together form a particular niche, contributions from a work that addresses both themes may benefit a larger audience.

In most studies, the participants carried out the gestures while standing [30, 34, 53] or information on posture is not clear [54, 33, 42, 36, 55]. Nevertheless, a few studies explored standing and/or sitting [37, 31, 40]. When comparing results for standing and sitting, Luna and coworkers found that performance is better when standing (hit rate ranging between 82% and 97% vs 65% and 80%, for 6 gestures, using accelerometer data from 15 subjects and multi-dimensional dynamic time warping). Nevertheless, only one work has been found that proposed tackling the recognition of gestures while laying in bed. Lamb et al. have used gestures in a bed scenario to allow automatic bed positioning using gestures [56], but the initial work developed has not only used computer vision methods to detect hand gestures, which is a considerable disadvantage compared to wearable sensor solutions, but it is also not clear if the gestures recorded were taken when laying in bed, so the results have to be taken carefully when trying to compare approaches.

When considering contributions related to communication support not involving sign language, those are rather infrequent. Exceptions are the works on speech generation using hand gestures [57] and interaction improvement for people with motor and speech impairments [58]. However, they are not suitable for the scenario being addressed for this dissertation since they rely, once again, on computer vision methods, with the use of a Leap Motion (infrared camera) and RGB camera. Research into in-bed situations including wearables has mostly focused on human activity monitoring (e.g., to monitor sleep quality, alert carers, or trigger automated actions). In this regard, it is critical, in my opinion, to give an extra and important degree of support for a wide variety of people (e.g., the elderly) in the bed setting, going beyond monitoring, by integrating sensors in a bedroom context to collect and apply data to enhance communication in that scenario.

In the context of the in-bed situation, it is critical that gestures, unlike sign language, be simple to execute and understand, taking into account the setting and the population being addressed. It is also worth noting that, while aphasics are the primary focus of research, developing a system that works for all sorts of aphasics would be challenging. Because aphasics might have complicated comprehension problems, the goal is to help individuals who still have some of those characteristics unaffected.

This overview highlights a good amount of work developed on the area of gesture recognition. Although many studies continue to unveil important results in this area, it is important to not only address different scenarios (e.g., the bed or bedroom scenario) but also to use those results to develop clever solutions to support people in their needs and to, for instance, effectively provide new interactions with the real world.

Chapter 3

Scenarios and Requirements

The following chapter will perform a detailed study of the people who will be impacted by this project to describe the intended system's users, as well as their needs. To accomplish so, fictitious personas and scenarios involving the personas and the system were created in order to assess and identify the system's needs, allowing a better development of a system.

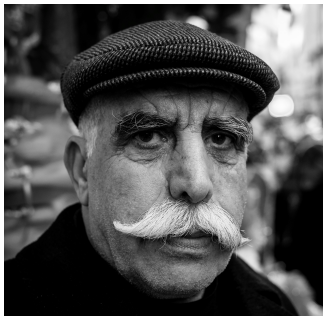
3.1 Personas

Personas are fictitious characters that are created based on research to represent the many sorts of users that could use some service, product, website, or brand in a similar way [59]. Personas and their motivations help not only in the formulation of system scenarios but also in the establishment of system requirements, giving real-world goals for the final solution.

Considering the scenario that the system aims to attain, there is a common characteristic among all personas going to be described next - **having aphasia**. Although this may limit the characteristics of those being described, there are still some different characters with different needs, expectations and capabilities, that may be envisioned for the system.

In this regard, an analysis of the literature about the categorization of different forms of aphasia and individuals impacted by it was done to gather the required information for the definition of the personas.

António Andrade



António Andrade is a sixty-two-year-old divorced man, currently living with his son on a two-story flat in Póvoa de Varzim, Portugal. António is a very skillful man, plays various instruments, and even restores some to give to a school in his neighborhood.
António suffered very recently a stroke and started having some trouble expressing his thoughts and understanding words, because of that he was diagnosed with Primary Progressive Aphasia (PPA), known for gradually affecting the communication skills of the patient.
Even though he was diagnosed with aphasia, he is currently in a stage where he is still active and takes part in daily life activities, usually leaving the house by himself to go shopping and to visit the school to which he donates some instruments because he enjoys the interaction with a younger audience.

Motivation: From the day his dad moved in with him, he had some trouble supplying privacy to his dad and, for that reason, wanted a way for him to have the chance to call him if anything happens, a simpler way that would deny the need to go to his room during the night without the need of.

Maria Julia, Retired

Maria Julia is a seventy-two-year-old mother of one and grandmother of two. She currently lives with his son, his wife, and both their kids that come home every weekend from college. She was married but her husband died ten years ago from health complications related to smoking.
Julia used to live alone with his husband, but since his death and being diagnosed with aphasia due to a cerebrovascular accident (CVA) 2 years prior, she moved in with his son so he could assist and provide her a safer and healthier environment by having more people around.
Julia worked in a hair salon her entire life so she was used to being around a lot of people, especially women. Her husband was very special to her and since his death, she started developing some signs of depression that led her to be accompanied by a psychiatric doctor. These signs have gotten worse since the cerebrovascular accident that she suffered because it led to developing hemiparesis that leaves her insecure.
She never had contact with a smartphone or smart gadget in general, so her knowledge at that level is restricted. Due to the stroke and suffering from aphasia and hemiparesis, she has trouble getting in and out of bed and has some acute pain during her sleep that may arise as an emergency at any time.

Motivation: Because Julia's bedroom is on a different floor from his son's, she needs a way to communicate her will, giving their son and family more confidence and trust that she is safe.

Adam Will, Shop Attendant

Adam is a 43-year-old, single man, living by himself in Ponte da Barca, Viana do Castelo, Portugal. For the past 20 years, he works in the same local shop being an attendant and often does volunteer work where he lives. Besides his daily work, he enjoys taking care of his garden, having a good selection of uncommon lily species that he sells online.

Two months prior, he suffered a stroke that led to the development of aphasia. He is currently only affected slightly in object nomination, so for the moment, he still keeps his job. Although he is currently fine, because he lives by himself, a caregiver was assigned to him to allow him to have some point of contact in an emergency.

Motivation: Owing to the fact that the caregiver is not present every night in his house he needs a way to communicate any emergency or need without the use of his voice, but rather with gestures that support the natural evolution of its condition.

3.2 Scenario

3.2.1 Scenario 1: Urgent Medication Assistance

Júlia, in her sleep, wakes up and feels an acute pain that makes her have to call her son to administer some urgent medication. She decides to **execute the gesture of come** [→ *REQ*2], so that her son is alerted. The system detects the gesture correctly and **sends a notification to her son's smartphone** [→ *REQ*2] that receives an indication of a potential problem. To confirm to his mom that the message was received and that he is coming, he may **send a message of confirmation** [→ *REQ*5] that is broadcast to his mom through a speaker in her room, giving more confidence to her.

3.2.2 Scenario 2: Home Safety Hazard

António had a rough day and fell asleep early. Suddenly, during the night, António gets' woken up by a strange sense of dizziness and a strange smell. He quickly realizes that the smell is actually of propane and **urges to get up, yet he fails to succeed** [→ *REQ*1]. To get help in this situation, he uses the APH-Alarm system to alarm his son of the situation happening in their home, by **executing the gesture of knocking on the bed mattress** [→ *REQ*2] to alert to a dangerous situation. His son then **receives the alarm** [→ *REQ*4] and quickly understands the severity of the situation calling the firefighters immediately and **sending a confirmation to the caregiver** [→ *REQ*6].

3.2.3 Scenario 3: Anxiety

During a nap after lunch, Adam starts to feel acute pain, **preventing him from standing up** [→ *REQ*2] and causing him anxiety. To get help, he executes the **gesture of twisting his wrist** [→ *REQ*1], which activates the APH-Alarm system and **alerts his caregivers** [→ *REQ*3] of a potentially problematic situation.

The caregiver receives the warning via the **caregiver app on his/her smartphone** [→ *REQ*4] and has the **choice of asking** [→ *REQ*5] Adam a few basic yes or no questions (including: Need water? Need medication? Is it an emergency?) In order to assess the situation's severity or gather extra information.

These **questions are broadcast to Adam through the speaker in his bedroom** [→ *REQ*6], which he will answer by using the predefined gestures supported by the system.

3.3 Requirements

Although different methodologies allow different requirement assessment processes, most requirements should be set in the early stages of the development cycle of a system to allow a successful run of development.

There are two types of requirements that may be defined, functional and non-functional. Functional requirements are requirements mostly related to features of the system, while the non-functional can be described as being attributes of the system such as performance.

Given the importance of the system's implementation and the fact that it is the initial iteration, the requirements were chosen to assure the proposed system's proper development and to provide the groundwork for future improvements and additions. Therefore, in this first iteration, the requirements are based on the needs to accomplish the scenarios proposed.

The non-functional requirements of the system are the following:

1. **Suitable for the bedroom:** The system should be suitable and easily integrated with the bedroom scenario, namely the specific in-bed scenario with a user lying in bed;
2. **Privacy:** Preserves the privacy of the user and safety of data;
3. **All-day use:** Should allow the use of it during all parts of the day, as long as the person is lying down in their bedroom;
4. **Provide communication means:** The system should support communication based on a pre-defined set of dynamic arm movements as input and provide speech output;
5. **Easy to comprehend:** The gestures used for communication need to be easily executable by everyone in the considered scenario, and they should take advantage of the context as much as possible by including the usage of the bed's mattress or other components of the bed.
6. **Performance:** The system should have a good performance and react quickly, in less than 5 seconds, preferably lower, upon interactions by the user and also on the process of sending and receiving information (*i.e.* between user and caregiver).

The system's functional requirements are sorted by importance and are included in the Table 3.1, along with parts of the scenarios to which they apply, and a brief description.

REQ	part of scenario	Requirement
1	"preventing him from standing up"	The system has to support users whenever they are laying in bed, with continuous operation regardless of the time of day or night.
2	"execute the gesture of come"	The system needs to correctly detect a pre-defined set of gestures, minimizing false positives as much as possible.
3	"alerts his caregivers"	The system needs to generate alerts based on the recognized gestures and send them in form of notification to an application running on the smartphone of the user's defined contact(s).
4	"caregiver app on his/her smartphone"	A caregiver smartphone application, to be in the possession of the caregiver, should receive the alerts from any communication attempts, as quickly as possible.
5	"choice of asking"	The caregiver app should be able to send Yes/No questions while also receiving the responses from the user (*i.e.*, gestures can also be used to answer questions).
6	"questions are broadcast to Adam through the speaker in his bedroom"	The setup needs to contain a speaker that provides a speech output modality to the system. This modality is responsible for broadcasting both questions from the caregiver or the alert that the message was received by the caregiver.

Table 3.1: Functional requirements of the system sorted in order of importance with a description and part of the scenario where they are considered.

Chapter 4

System Proposal

This chapter presents the steps involved in the development of the gesture recognition system developed to support people with speech impairments, mostly aphasics, through the recognition of gestures. The chapter is divided into 3 sections. The **first** presents a high-level overview of the solution implemented. The **second** section presents the responsibilities of modules and their interactions as well as some details on the choices of implementation. The **third** and final section presents in-depth details of the implementation of the modules.

4.1 System Architecture

The system is described in this dissertation aims at providing communication support to people with speech disabilities, mainly aphasics.

People with speech and language disorders, such as aphasia or others, often experience difficulties in expressing their needs in a way that can be understood by others. These difficulties cause major limitations to their independence, so body movement, gestures, and pose can potentially play a role as alternative ways to communicate.

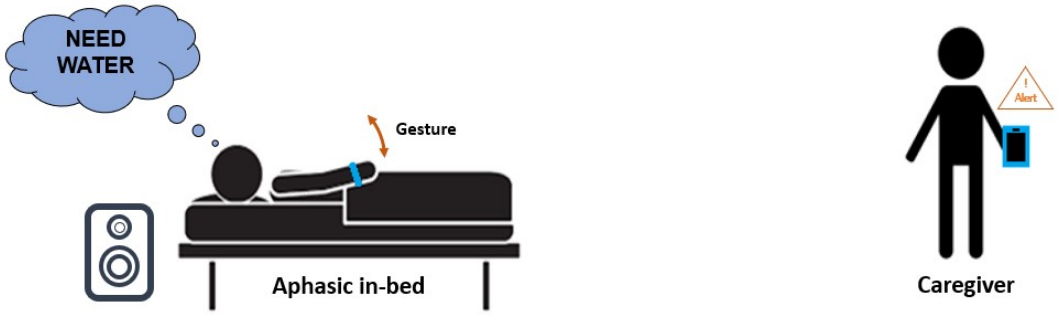


Figure 4.1: Basic scenario depicting the goal of the system.

One of the situations where they can feel more vulnerable is their bedroom (Figure 4.1), especially when laying in bed. With that in mind, the target of our system is the bedroom environment, where a person is alone and lying on a bed, and may need to communicate with other people (e.g., to ask for water), while resting during the day or awake at night.

The goal is to provide not a monitoring solution but a way for these users to communicate with a caregiver, family, or friend.

To accomplish the goal proposed, the bed scenario had to be reimagined to detect movements and other signals. For that, it was considered the use of sensors attached to the body and/or mounted to the bed that connects, preferably wirelessly, to a processing unit that will be able to perform all steps from collection to feature extraction and classification of the movements. If anything relevant is detected the processing unit should be able to decide and send a notification to the caregiver responsible for that person.

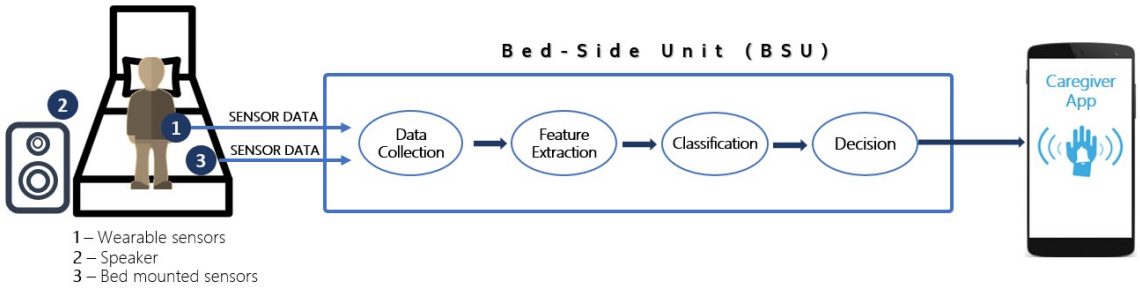


Figure 4.2: Basic architecture depicting how the bedroom scenario was reimagined.

Figure 4.2 aims at providing a high-level overview of the bedroom scenario just described. The (aphasic) user is laying in bed, with sensors both placed on the user's dominant wrist and the bed. At any time, he decides to alert his caregiver of an event by executing a movement. Data from the execution of the movement is sent to a local bedside unit that is responsible to identify the relevant movements executed. If any relevant one is detected, the bedside unit should be able to notify the caregiver through a smartphone notification sent to the caregiver app present on this smartphone. Not only that, but the caregiver may also opt to send a confirmation of the alert reception, as well as simple yes/no questions that can be replied to using movements by the user.

The method just described allows an aphasic to call a caregiver who may not be present at all times in case of an emergency, providing a sense of warmth and protection to both the aphasic and the caregiver.

4.2 Modules Interaction

The system being implemented has three main modules to consider: the **sensors** responsible for collecting data, the **bed-side unit** responsible for aggregating data, classifying movements, and the **communication module**, responsible for employing the means for every service to communicate.

4.2.1 Sensor Layer

Sensors are devices that respond to changes in the environment reporting the value, usually, to another device. Sensors can be used in a plethora of scenarios, but when considering the bed scenario most uses for them include monitoring of some sort, for instance, monitoring sleep posture [60], monitoring blood pressure [61], respiratory rate monitoring [62], mostly using bed-mounted sensors. Although bio-sensors are commonly used for health monitoring, our system is focused on activity tracking, specifically quantifying arm movements that can be associated with meaningful gestures.

Wearables containing accelerometers and gyroscopes, as well as in-bed solutions using them, are a great way to detect movement on the body's extremities and even elements of the bed. The downside of applying in-bed solutions is that they may interfere with the bed environment and cause undesirable alterations to the bed scenario, resulting in an unpleasant setting.

For this system, a wearable is chosen to offer a less invasive but yet reliable method to detect movement using classic sensors. This wearable is not as obtrusive as one might assume because it is akin to the use of SOS panic buttons that many seniors are familiar with.

This wearable will be used on the user's dominant wrist and, through the integrated sensors will be able to report the signal variations that may be processed externally to detect if any relevant movement was executed. With that goal, the wearable will be responsible for retrieving sensors data and sending it to a specialized device for processing. To do that, the wearable has to establish a channel of communication with an external bedside unit, transferring the responsibility of movement classification from the device with the sensors to that unit.

4.2.2 Bed-Side Unit

The bedside unit is responsible for the aggregation of the sensorial data received by the sensors and the classification of the data to alert a caregiver if any relevant movement is performed. This unit is also responsible for implementing a speech output service that will aid the user in the obtainment of messages from a caregiver.

The bedside unit should have enough computing power to not introduce significant delays on all steps involved in the processing and classification of the data, and should also be able to establish a reliable connection with the sensors.

Figure 4.3: Classification pipeline used to classify movements performed.

Figure 4.3, presents the processes involved in the classification of the movements. The steps include **data collection**, responsible for receiving the data from the sensors, **feature extraction**, responsible for extracting relevant features from the sensor data, and the **classification**, which takes the features extracted and classifies accordingly to the predefined movements supported by the system.

To implement the speech output service that converts text messages to speech using text-to-speech (TTS) technology and plays it using a speaker, the service has to retrieve messages sent by a caregiver by providing some channel of communication between both parts.

4.2.3 Communication and Interaction

To implement the speech output service that converts text messages to speech using text-to-speech (TTS), as well as the service that allows the sending of alerts to the caregiver, the bed-side unit has to establish a bidirectional channel of communication between both, the user, through the bed-side unit placed in the bed environment, and the caregiver, through a mobile application that may be used everywhere as long as there is an active internet

connection to receive messages and send messages. To allow this, an interaction manager is used which provides endpoints to other external services, devices, or others, to connect and send information to the system.

To allow the caregiver to receive messages, the smartphone must have an active internet connection which allows the smartphone to receive notifications and therefore alert him for the execution of a movement. This implies that the bedside unit must also have an active internet connection to allow sending notifications as well as receiving questions by the caregiver, thus providing a bidirectional channel of communication between both parts.

4.3 System Implementation

This section will provide an in-depth description of the implementation of the different modules and services provided by each of them. The implemented system relies on the use of classic sensors to detect movement, while the use of in-bed sensors is not considered for this prototype.

4.3.1 Deployment View

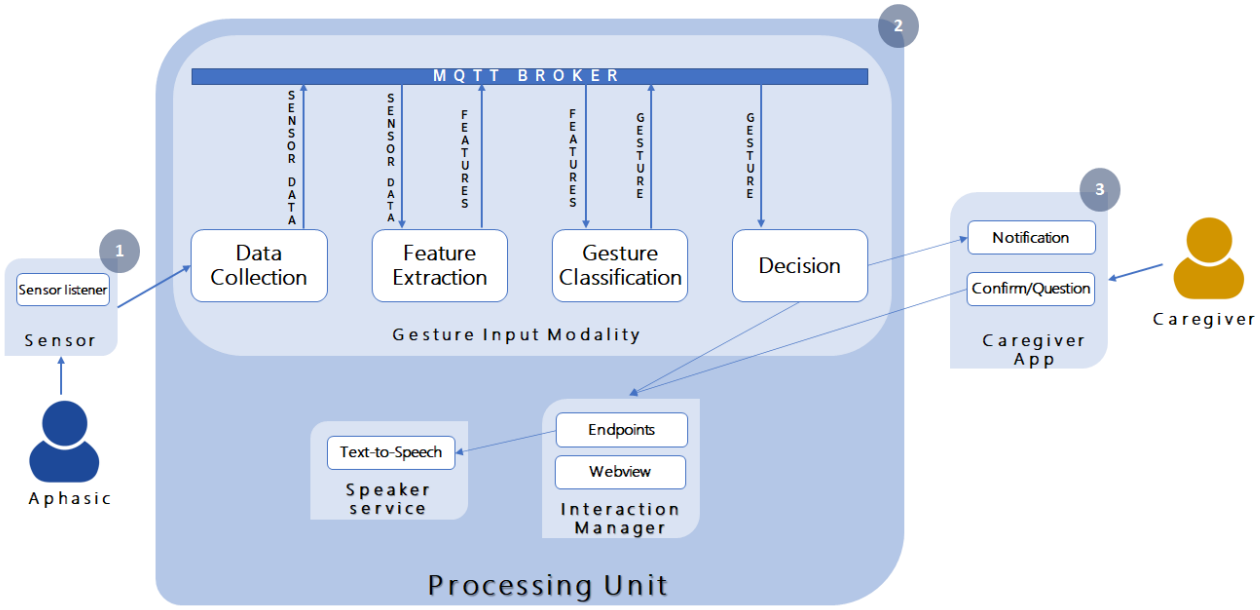


Figure 4.4: Diagram with the services and interactions between them (1 - Wear OS Smartwatch; 2 - Raspberry Pi 4; 3 - Android Smartphone).

The system is implemented using a modular design, in which each module is constructed independently of the others and implements a component of the system with a well-defined interface. This choice brings numerous advantages especially when considering code update or restructure of the system. Although this may bring additional latency compared to a monolithic approach that includes all services in a single program, the advantages in terms of modularity and scalability outweigh the disadvantages [63].

For the smooth operation of the complete system, the solution, as stated, is segmented into distinct modules that operate independently of one another and may be replaced with others without restructuring the system as a whole. These modules are listed and briefly described below:

1. **Sensor data recording**, an application developed for the Wear OS smartwatch that continuously reads sensor data and sends it to the processing unit using Bluetooth.
2. **Gesture recognition service**
   (a) **Sensor data collection**, advertises a Bluetooth service that serves as the endpoint for the smartwatch to communicate and publishes the sensor data.
   (b) **Feature extraction**, from the data, received the important features are extracted to be used for the classification in the following stages.
   (c) **Gesture classification**, a previously trained model is used to classify the movements done and identify gestures executed.
   (d) **Decision**, based on the classification results there needs to be a decision on what to do, either send a message to the caregiver responsible or not.
3. **Speaker service**, service responsible for receiving messages from the caregiver and broadcasting it through the speakers using text-to-speech.
4. **Monitoring and configuration services**, a web service developed to enable further interactions with the system. This service implements a web view that allows data visualization of the classification of the movements being executed, changing of the model being used for classification, and implements endpoints used for interaction with the system.
5. **Caregiver app**, an application developed to be used by the caregiver to receive the alerts and send messages to the aphasic to his responsibility.

The different modules communicate using either HTTP Communication or Message Communication, subject to the use of a message broker. These modules will be discussed in further detail in the following topics, including specifics on the processes and judgments made.

To deploy the solution, a set of devices were chosen. The smartphone used to test the caregiver app has the only constraint of being an Android smartphone, and for this dissertation, a Poco F3, with Android 11 was used. The speaker of choice is a generic, no-brand speaker with an auxiliary input. The smartwatch and the processing unit, which are more relevant to the project and have certain significant specifications, are discussed next.

Smartwatch

For this dissertation, an OPPO Smartwatch with Wear OS was chosen. Although other options for smartwatch's exist, this watch was chosen for its features and specifications, which are presented in Table 4.1.

Having Wear OS was the main point. Wear OS is a version of Google's Android operating system designed specifically for smartwatches and other wearable devices. Wear OS includes a variety of connectivity options, including Bluetooth and WiFi, as well as several useful features and applications [64].

Specification	Description
Operating System	Android Wear OS
Processor	Qualcomm Snapdragon Wear™ 3100
Memory	1GB + 8GB
Batery	300mAh
Connectivity	WiFi, Bluetooth 4.2, Bluetooth Low Energy, GPS, NFC
Misc.	Heart Rate Sensor, Tri-axial Accelerometer, Geomagnetic sensor, Barometric

Table 4.1: OPPO Smartwatch main specifications.

One of the many advantages of Wear OS is that it allows developers to write programs in Java or Kotlin in the same fashion that they would on an Android phone, allowing them to, for instance, listen to the smartwatch's sensors and communicate with other devices. For that reason, having the ease of developing in Android improves the development of applications but also allows porting the work done in one type of device to another, with the same operating system. Also, the sensors present in the smartwatch were on par with the desired set of sensors for the project.

Figure 4.5: Oppo smartwatch.

A Wear OS app was developed to read the sensor data from the in-built sensors and was also responsible for establishing a connection with a processing unit to send data.

Processing unit

A Raspberry Pi 4 was chosen as the processing unit for the system. As stated, the processing unit is responsible for connecting, collecting, and processing the sensor's data to classify the movements of the user into relevant information.

Figure 4.6: Raspberry Pi 4 Model B.

A Raspberry Pi is a compact single-board computer that was designed to teach basic computer science in impoverished countries' schools [65]. This compact single-board computer includes an ARM CPU and thus has Linux compatibility, making it a great fit for projects like the one being developed here, where it will serve as a connection point for sensors and other devices. The specifications for the model chosen are found in Table 4.2.

This device must be able to maintain a solid connection with the sensors without causing significant latency, allowing for quick recognition and therefore a reliable overall system. This unit should also be able to connect to other devices on-demand, for possible future improvements of the overall system, with the introduction, for instance, of new sensors or devices that allow further interactions.

Specification	Description
Operating System	Raspberry Pi OS
Processor	Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
Memory	64Gb + 8GB LPDDR4-3200 SDRAM
Charging	5V DC via USB-C connector (minimum 3A*)
Connectivity	2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE

Table 4.2: Raspberry Pi 4 specifications

4.3.2 Sensor Data Acquisition

The **sensor data recording** service is directly deployed on the wearable. As previously mentioned, a Wear OS smartwatch was chosen for this dissertation owing to many advantages, one of which is that the system uses Android and its development platform, making the process of developing new applications for various purposes easier.

The application, deployed on the smartwatch, implements two important methods:

1. **Sensor listener**, that retrieves and makes the sensor data available to be transferred;
2. **Bluetooth service**, responsible to establish a connection to a Bluetooth server deployed on the processing unit and send periodic messages to it with the sensors data;

Both methods will be discussed thoroughly next.

Bluetooth service

For the smartwatch to establish a connection to a Bluetooth server deployed on the processing unit, a Bluetooth service needs to be deployed on the smartwatch that is capable of establishing a channel of communication between both devices.

To implement this Bluetooth service, Android utilizes **BluetoothDevice** to establish a **BluetoothSocket**, from which the thread may connect and start a communication channel.

In order for a **BluetoothDevice** to connect to **BluetoothSocket**, the smartwatch must search for a server that advertises a service with a shared UUID between them. If the lookup is successful and the remote device accepts the connection, a channel of communication is created and may be utilized to share information. This channel can be used for both reading and writing, although it's mostly used to send values from the smartwatch to the processing unit.

Sensor listener

Nowadays, most Android devices have integrated sensors that allow the retrieval of raw sensorial data with high precision and accuracy. Due to the increasing performance of those devices, similar scenarios to the one being addressed in this dissertation started using Android devices to collect data. Although some other sensors are available in the smartwatch used, only the accelerometer, gyroscope, and magnetometer are relevant for our scenario and only those are actively read.

To access the raw data from sensors on an Android device, the Android sensor framework is used. This framework enables, among other things, the identification of sensors and sensor capabilities, as well as the monitoring of sensor events, which occur whenever the sensor's data changes.

To allow the monitoring of sensor events, the thread responsible for that needs to implement two methods:

1. **OnAccuracyChanged()** - invoked every time the accuracy of a sensor changes.
2. **OnSensorChanged()** - which is invoked every time the sensor detects a new value and contains not only the new value of the sensor as well as a timestamp and the accuracy.

When registering a sensor listener, it is not possible to define the polling rate of the sensors, nevertheless, it may be estimated by the timestamps of the data recordings, but it is not feasible to use the framework to specify a precise polling rate and therefore, external methods need to be employed to obtain one.

Java provides mechanisms to schedule threads to run at a specific frequency. Because there is a need to constantly provide data at a fixed rate, an executor together with a scheduler was created and used. An executor allows to schedule commands to run after a given delay, or to execute periodically [66], to execute periodically a time frame has to be provided and the scheduler takes responsibility for providing the periodicity.

For this system, a fifty-hertz polling rate was chosen. Having a fifty-hertz polling rate means that every twenty milliseconds a data value needs to be recorded, for that reason the scheduler was set to that rate, and every twenty milliseconds the value reported by the sensors is recorded together with the timestamp of the creation of that segment.

After fifty segments have been recorded (one every twenty milliseconds, totaling one second), each with a value for the accelerometer, gyroscope, magnetometer, and a timestamp, the data is transferred to the processing unit via the channel created by the Bluetooth service for that purpose.

This data segment is delivered together with a packet counter for synchronization reasons and a timestamp to enable delay monitoring, using JSON format as presented below.

```
0 {"data_segment": {
1   "packet_counter": 0,
2   "data": "acc_x, acc_y... 0.2, 0.7, 2.5,...",
3   "timestamp" : 1632259181
4 }}
```

4.3.3 Sensor Data Collection

The Sensor data collection service, implemented on the processing unit, is responsible for advertising a Bluetooth server to which the smartwatch connects and establishes a channel of communication.

When it starts, it has the responsibility of advertising a service that the smartwatch or other sensor device can understand as being the server for data collection, which can be achieved by using a UUID that is shared among the devices.

Although other programming languages may be used to implement a Bluetooth server, Python 3 (version 3.7.10) was chosen. Bluetooth communication is supported by python's native sockets but since it is easier to implement, the "PyBluez" (version 0.23) module was used for this purpose. This module allows users to deploy Bluetooth servers/clients, ranging from basic to more sophisticated, starting with very simple client and server programs and

progressing to coping with unreliable servers and even providing Bluetooth Low Energy compatibility (BLE) support. This is all very well documented and examples for some scenarios are presented in their repository and were used for this project [67].

In this service, the flow can be briefly described as follows:

1. Service starts a Bluetooth Socket that accepts connections;
2. A Bluetooth server with a unique "UUID" is advertised and other devices may connect.
3. When a device tries to connect using the same "UUID" a channel of communication gets established and data may be transferred.
4. Data is received in 1024-bit trunks, which implies that, if the message is bigger than that, it cannot be received all at once. As a result, data is concatenated from the beginning of the message until the end is identified, this is possible by using a "JSON" format on the messages and detecting the terminator.
5. When a full message is received, the sensors data is extracted and gets published to an MQTT Message Queue that other services can subscribe, consume and process the data.

4.3.4 Feature Extraction

The **Feature Extraction** service is responsible for extracting the important features from the sensor data received for it to be used for classification. Selecting the most appropriate features is a critical step in data analysis because it allows to process information quicker, reduce model training time, and determine which features are more significant to the problem being solved.

This service has the responsibility of consuming and processing the sensor data retrieved from the message broker used by the last service, extracting the important features needed to classify the motion as one of the supported gestures.

Since this queue receives a message approximately every second, it is critical that the feature extraction done does not take too long, causing unwanted delays. Although once again, any programming language could be used to consume and process the data, python was once again chosen due to the data analysis capabilities provided by the "pandas" (version 1.2.4) module.

The features extracted from the sensor data are listed in Table 4.3.

Feature	Description	# Features
Mean	Average of each signal axis	9
Variance	Variance of the signal	9
Median	Median of each signal axis	9
Range	Range of the signal	9
Standard Deviation	Deviation of each signal axis	9
Skewness	Measure of asymmetry of the probability distribution of the signal about its mean	9
Kurtosis	How peaked the sensor signal distribution is	9
Correlation	Pearson's correlation between each pair of axes (xy, yz, and xz)	9
Integral	Area under the curve	9
Sum of all squares	Sum of the squared value of all samples for all three axes of a given sensor	3

Table 4.3: Time-domain features extracted for each sliding window.

After the features are extracted from the data retrieved, a single segment of data with the important features is available to be used for classification. This segment is therefore published into an MQTT Message Queue to allow the service with classification purposes, or others, to consume it.

4.3.5 Gesture Classification

The **Gesture Classification** service has the purpose of consuming the MQTT Message Queue containing the segments of data published by the service described in **Subsection 4.3.4**. With the values retrieved, they are used to classify the movement executed.

To allow classification, research into the topic of gesture recognition was addressed and an in-depth analysis of the topic is addressed in Section 5. From the results obtained, the best combination of classifiers and other relevant variables was determined and used in the system implemented, more specifically in the service being described here. The investigation done, allowed not only to address the best combination of variables for classification but also allowed the attainment of a trained model, with data from ten participants that demonstrated promisingly good results considering a subject independence scenario, which is relevant considering a real-world scenario.

In essence, this service consumes a segment with the important features extracted and uses a pre-trained model to classify that segment into a gesture. The model used for classification can be either the one saved locally on the processing unit or the one in a remote server like the service described in 4.3.8, that also retrieves the gesture executed.

The result of the classification is given as a number from one to six. From one to five are the five gestures supported by the system, and the sixth is the "No Gesture", meaning no gesture of the ones supported was executed. The number retrieved from the classification is used later to assess if a gesture was executed and a message needs to be sent to a caregiver/family/friend.

Similar to the other services, the results from the classification are published into an MQTT Message Queue where other services may retrieve the data and process it arbitrarily.

4.3.6 Decision

The **Decision service**, the last service provided by the gesture modality is a decision on what to do with the classification information. It is not a wise option to believe that every classification is guaranteed to be correct because the model chosen does not have perfect accuracy for the classification of gestures. As a result, the decision service will only conclude that a user is executing that exact gesture if three consecutive windows of classification terminate with the same gesture.

When three consecutive windows result in the same gesture the system assures that the user executed a gesture and a message may be sent to the caregiver indicating that help is needed. On top of sending a message to the caregiver, this service uses one of the endpoints provided by the web service developed to publish a gesture executed together with a timestamp indicating the time of execution of it.

4.3.7 Speaker Service

The Speaker service provides audio output to the system. It has one major application: reporting on questions or confirmations sent to the caregiver app by the person responsible for a specific user.

This service relies on a physical speaker, connected directly to the processing unit and a thread that is constantly waiting for questions or confirmations to be broadcast. When a message is received, the service is responsible for broadcasting it through the speaker on the system using Text-to-speech (TTS) technology, allowing the user to listen to the question without the need to look at the screen.

To accomplish this, the Python module "Pyttsx3" (version 2.90) is used and is responsible to broadcast the messages. This module implements Text-To-Speech conversion fully offline, and allows to change a few parameters such as **volume**, **voice** (male, female), **speech rate**, among other features. The only parameter that was adjusted from its default settings was the **speech rate**, which was reduced to allow for a better understanding of the questions and confirmations, keeping in mind that the system is aimed at a group with certain speech impairments who are mostly elderly.

4.3.8 Monitoring and Configuration Services

To allow the monitoring of the system and the possibility of configuring certain features of it, a web service written in Flask (version 2.0.1) was developed to support the solution in place and offer more interaction with the system. This web service includes endpoints that allow interaction with the system, and a web view that allows including but is not limited to, visualizing through a plot of the last two minutes, the result of the classification of the movements that are being performed,

In short, the system allows to:

1. Visualize a plot with the evolution of gestures being executed by the user associated (Figure 4.7);
2. Configure the model used for classification in real-time (Figure 4.8).;
3. Allows to send a dataset with the correct features calculated to be trained and used for classification;
4. Implements endpoints allowing further interactions between different parts of the system.

Figure 4.7: Plot showing the gesture being executed in real-time.

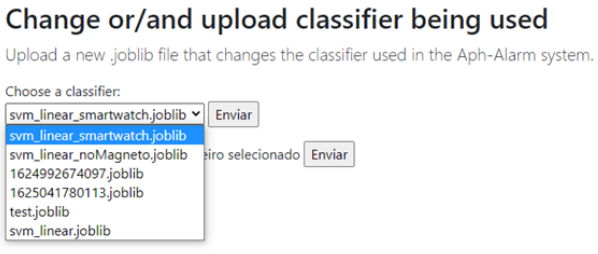


Figure 4.8: Change classifier feature provided by the web service.

To deploy this web service, as stated, a flask Python application was created. Flask is a web microframework for python and makes it easy to develop a simple and lightweight web service, depends on *Jinja* as a template engine but otherwise gives freedom to develop as intended. The web service deployed provides some endpoints allowing further interaction with the system by external and remote services such as the caregiver app. The available endpoints are listed and described below:

1. ***pub_gesture_executed*** - endpoint used by the processing unit to advertise the latest movement executed. This is called every second and saves the value in a structure with the one hundred and twenty latest values (corresponding to two minutes).

2. ***list_gestures_executed*** - endpoint used to retrieve the list with the last gestures executed to a maximum of 120 entries.

3. ***classify_movement*** - endpoint used by the processing unit to classify movements. It allowed sending the features already extracted to retrieve a classification done with a model loaded on the server.

4. ***change_classifier*** - an endpoint that allowed to change the classifier model used on the server.

5. ***message_speaker*** - endpoint used to send a message that needs to be broadcast on the user's side. The caregiver app 4.3.9is in charge of transmitting a message that is saved and accessible by the processing unit. The speaker service queries this endpoint to see if any messages are accessible, and when a message is consumed, the endpoint is cleansed.

6. ***upload_classifier*** - an endpoint that allowed the upload of a *.joblib* file to be saved on the server. This file contains a trained model that could be used for classification.

This web server implements the basics for the intended purpose of the current solution implemented but can easily be improved in later iterations.

4.3.9 Caregiver APP

The caregiver app consists of a mobile application, developed to be used by caregivers/family/others, responsible for an aphasic. This application was developed for Android smartphones only and has three main features:

1. Receive notifications on gestures executed by the user, provided by Firebase Cloud Messaging (FCM).

2. Return a confirmation message to the aphasic to provide positive feedback on the message's receipt.

3. Send predefined yes or no questions to the user and receive the responses in form of the gesture executed.

To fulfill the intended features the smartphone must have an active internet connection and the webserver implemented in Subsection 4.3.8 must be up and accessible.

If any gesture is detected, a system notification is sent by the processing unit to the caregiver mobile application using Firebase Cloud Messaging. When a notification is sent (*e.g.*, a gesture was performed), the app unlocks new interaction possibilities (Figure 4.9), where either a question or a confirmation may be sent to the aphasic (Figure 4.10).

If the caregiver's only intention is to confirm that the message was received, the caregiver may select "Send a confirmation" which communicates the option with the correct endpoint of the interaction service described. Ultimately it triggers the speaker service and announces a confirmation message through the speaker.

If the caregiver needs more information he may choose to send one of the predefined questions. For this prototype, only three questions were chosen to be supported and are based on the scenarios defined in Section 3.2 (Figure 4.10), and then waits for a response from the aphasic (Figure 4.11). This screen accesses the endpoint that retrieves the latest gesture executed and when a different gesture is executed, it activates further interactions with the app (Figure 4.11), these interactions are the possibility of sending a confirmation or a further question back to the aphasic.

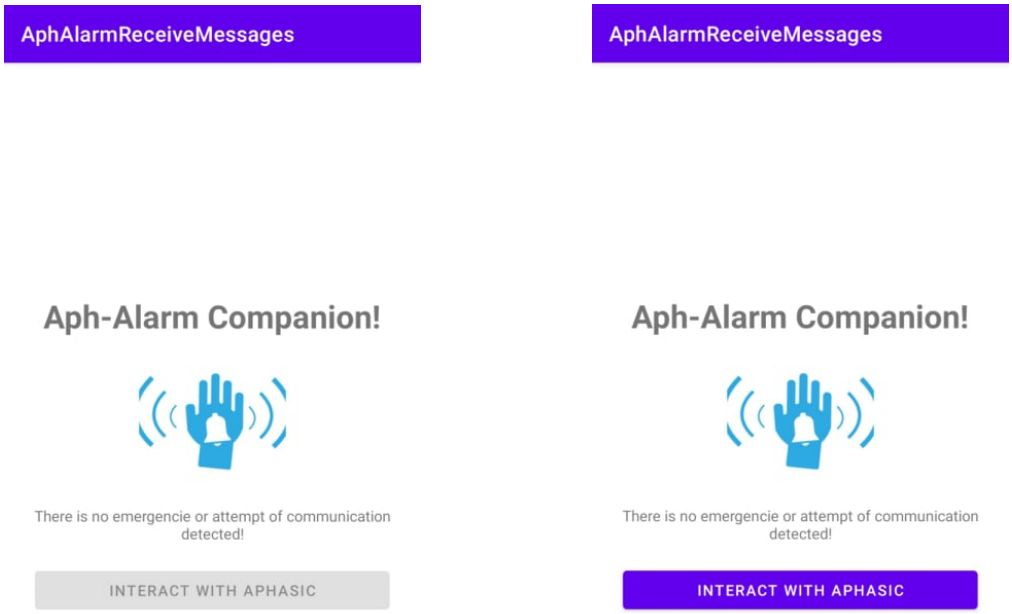


Figure 4.9: (Left Figure) Home screen. (Right Figure) Home screen when a gesture was detected and the system allows further interactions.

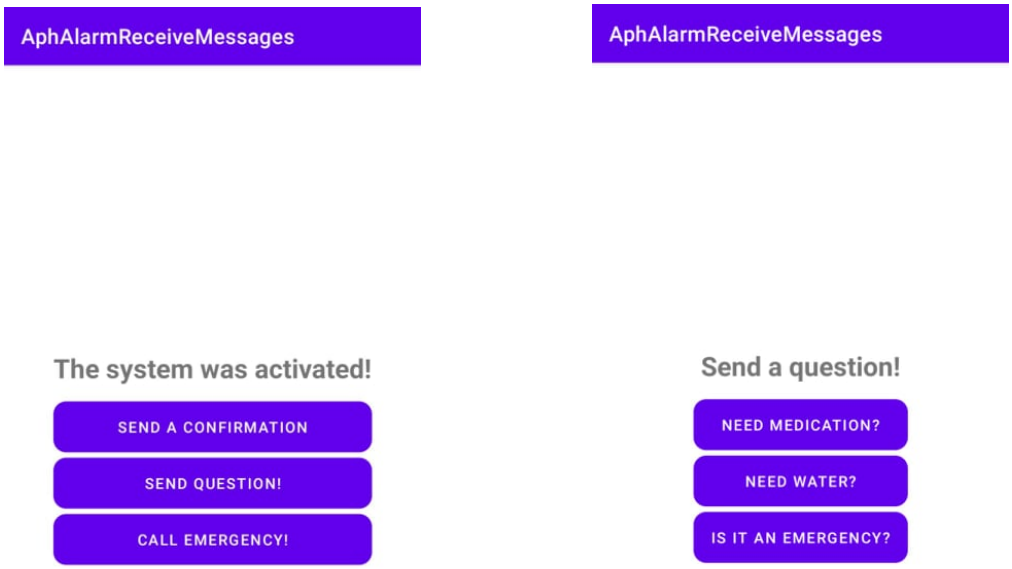


Figure 4.10: (Left Figure) Interaction choices available. (Right Figure) Predefined questions that may be used to interact with the aphasic.

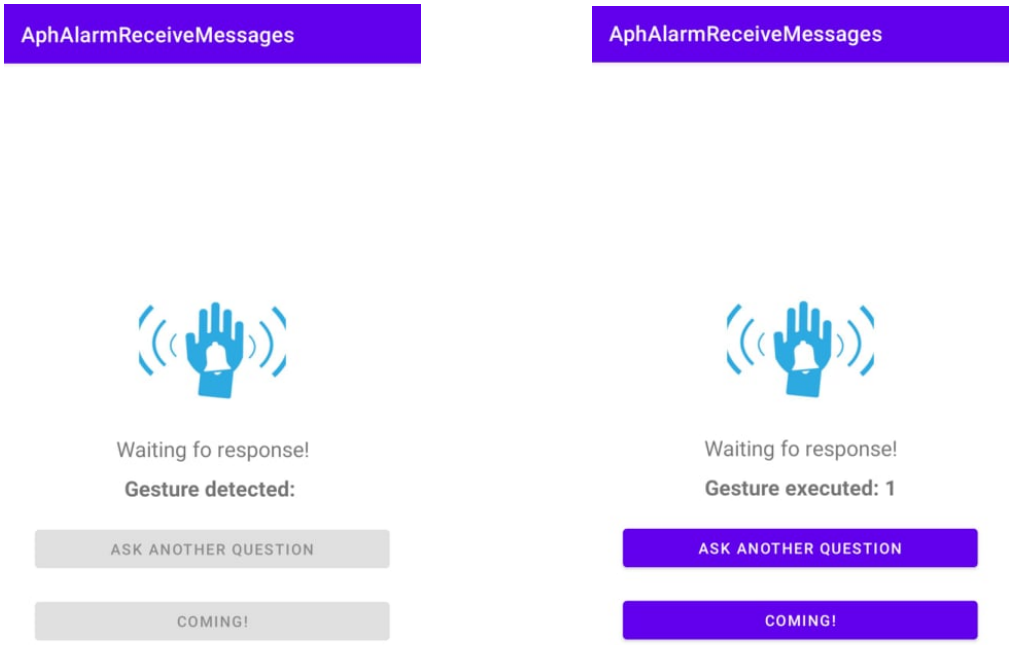


Figure 4.11: (Left Figure) Screen that waits for the response of the aphasic. (Right Figure) Screen that waits for the response with a response received.

Chapter 5

Results

The system proposed and described in this dissertation aims to support individuals with aphasia in communicating. It was defined that, to do it, communication would be based on the identification of a set of gestures that may have a certain defined meaning.

To accomplish this objective, two experimental procedures, one with a smartphone and another with a smartwatch, were carried out to gather data and understand the possibilities of obtaining a machine learning model good enough to accurately recognize gestures using only one wearable placed at the dominant wrist of the user. The effect of the use of different machine learning classifiers, sliding window sizes, and window overlaps was evaluated to find the best combination and obtain the best performing model in both a subject dependence and independence scenario, with the latter being preferred.

5.1 Gesture Selection

The gestures for the system implementation were chosen taking into account two crucial aspects: the bed scenario being addressed, and the target user group. The bed scenario implies that the user will be lying in bed and that the motions must be straightforward to comprehend and execute. Therefore, arm movements that make use of the mattress or other elements of the bed may be favored above others. The target population, which consists primarily of aphasics, has a past medical history of stroke, which has resulted in aphasia but is also likely to have resulted in other problems, physical or otherwise. This necessarily requires the use of dynamic arm movements that can be performed in a variety of ways, with varying speeds and amplitudes.

Gesture	Description
Knock	Knock with hand palm on the mattress, close to the body.
Twist	Twist the wrist, preferably, with a 0 to a 45-degree angle between the bed and forearm.
Clean	Move the hand from left to right and vice-versa, with the arm in contact with the top of the mattress.
Circle	Make a clockwise circle shape in the air, with the arm extended throughout the whole movement and an open hand, starting and ending closely at the same location.
Come (to me)	Move the forearm towards the arm, starting with the arm extended on the bed's mattress and ending with a 45-degree angle between the forearm and arm.

Table 5.1: Arm gestures considered for the system's prototype.

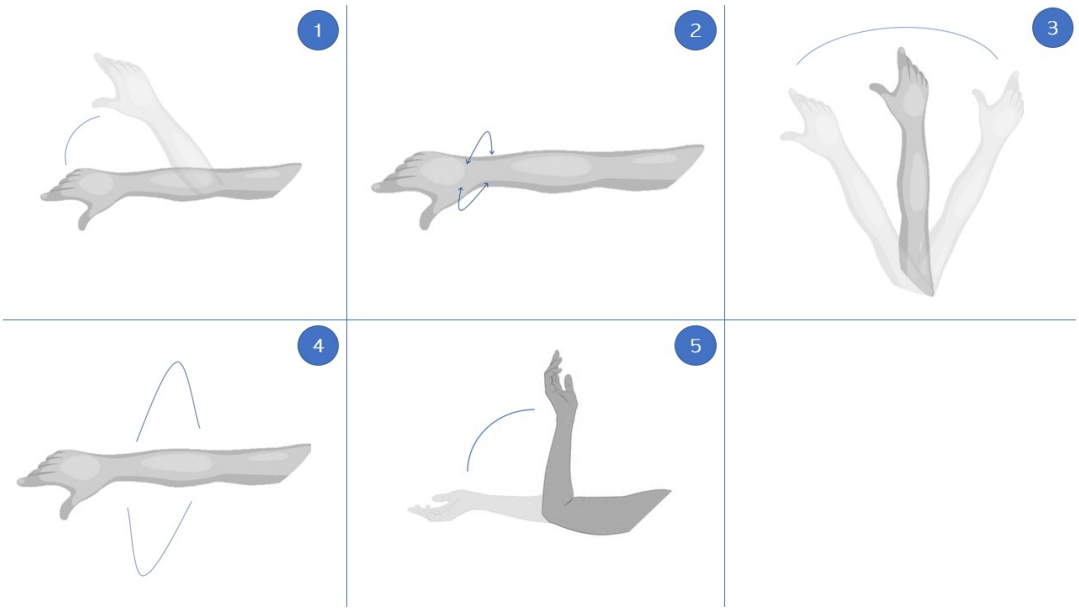


Figure 5.1: Set of gestures aimed to be supported by the system's prototype.

The gestures contemplated for the system being implemented are described in Table 5.1 and shown in Figure 5.1. To choose the gestures, state-of-art literature contemplating gesture recognition was used as a starting point. Although more gestures could be proposed, it was concluded that for the proposed system, although not optimal, a minimum of 2 gestures allows full interaction with the system (one to activate the system and send an alert to the caregiver, and two other to allow answering yes or no questions, one of which, in the worst scenario, may be the one used to trigger the alert.

Multiple works have used similar gestures such as up, down, left, right, circle, with good results using classical machine learning algorithms. Since those gestures can also be used in the bed scenario with some adaptation, the movement knock, which includes a movement of

up and down perpendicular to the mattress, the movement clean which encompasses left and right movement with the hand resting on the bed mattress, and the circle movement were included. Although present in fewer works, the wrist twist was chosen due to the ease of execution and the possibility of executing in different positions, from a wrist twist with the hand resting on the mattress to a high wrist twist. The fifth gesture is meant to be an intuitive gesture that allows the user to mimic calling someone. Although it had not previously been utilized in gesture recognition research, it was nevertheless chosen due to the meaning of the gesture for the considered context, keeping in mind the challenge at hand, which is providing new communication possibilities through the execution of dynamic hand or arm movements. After defining the gesture set, it was validated by a speech therapist with experience with aphasics.

To avoid detecting other arm movements, or the absence of movement, as an interaction attempt, a sixth gesture was considered, which is referred as to "No gesture". This "No gesture" intends to entail all the regular behavior that a user does while laying in bed. This includes all movement except the prescribed gestures and may include activities relevant to the bed scenario like body rotation or moving the hand towards the face.

5.2 Preliminary Evaluation

The first experiment for this dissertation was designed to acquire preliminary results on gesture recognition relying on a bed setup with only one sensor device attached to the user's wrist, to assess the possibility of progressing with the proposed setup.

With that target in mind, a first approach was developed using the built-in sensors of common smartphones as provisional movement quantification sensors (i.e, the user wore a smartphone on the wrist). Although not optimal, at the time, the availability of sensors was scarce, and to obtain early results, it was possible to replace the intended wearable with a common smartphone with similar sensors. The data collected were then processed to evaluate the possibility of using gestures in a bed environment. The processing involves four important stages shown in Figure 5.2, which will be discussed in greater detail below: **sensor data collection**, **preprocessing**, **feature extraction**, and **classification**.

Figure 5.2: Processing stream.

To evaluate this initial solution, two scenarios were considered: user-dependent and user-independent. Firstly, the user-dependent solution was conducted to explore the potential of different classifiers. For this scenario, different classifiers were explored with data from each user, one at a time, for both model training and testing. After that, the user-independent scenario was investigated by assessing the feasibility of having a solution where a model

trained with data from a group of subjects may be used to accurately classify movements from other, never-seen, subjects.

5.2.1 Experimental Setup and Protocol

The experimental setup included a prototype, which relies on a sensor device, including 3D accelerometer, gyroscope, and magnetometer sensors, attached to the user's dominant wrist. Due to some limitations, the device chosen was an Android smartphone [1], attached to the user's dominant wrist, containing the three listed sensors. The setup also included a bed where the participants lay down to perform the experimental procedure.

The initial evaluation of the prototype was carried out with gesture data acquired from three subjects. Table 5.2 presents some details of the participants involved in this experiment. Each participant was informed of the experiment beforehand, including the setup, the conditions, and the gestures that they would be asked to perform. They were also encouraged to practice the gestures, without the sensors, but this step was not critical to the experiment since the gestures were meant to be of ease of execution.

No. users	Sex (M/F)	Age	Weight	Height	Right/Left handed
3	1/2	32.67 [22, 54]	75.3 [60,83]	1,71 [1.62, 1.80]	3/0

Table 5.2: User characterization for the participants involved in the first experiment. The values for age, weight, and height are presented as mean [minimum, maximum].

Participants were instructed to position the wearable module on their dominant wrist and then lie down in the bed on their backs with the arms parallel to the body. In this position, they were asked to repeatedly execute each gesture listed in Table 5.1 for ten seconds each time. For this experiment, concerns regarding the order of execution of the gestures and the number of repetitions were not taken into consideration.

5.2.2 Dataset Characterization

Participant one provided most of the data, with a total of 17 executions for each gesture, while participants two and three contributed with, respectively, 4 and 2 executions per gesture. Only participant one contributed with "No Gesture" data.

Sensor data for each trial of a gesture were collected, labeled, and divided using a sliding window of one second with an overlap of 96% between consecutive windows. This sliding window size and overlap was the only combination explored in this first experiment, due mostly to the results presented in recent related work on gesture recognition using similar techniques [48, 49, 50, 46]. For each sliding window, a total of 84 time-domain features, presented in Table 4.3, were extracted and used for feature selection, which resulted in a dataset with a total of 29550 instances (24300 for participant 1, 3250 for participant 2, and 2000 for participant 3).

The plots presented in Fig. 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, are examples that show the sensor variations (accelerometer, gyroscope, and magnetometer) during the execution of the different performed gestures.

[1] Huawei PSmart with Android 8.0 EMUI 8.0 Oreo

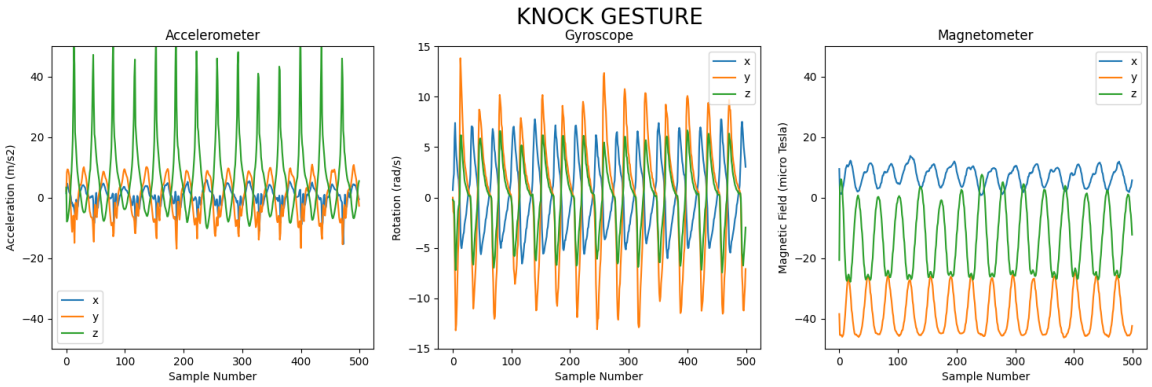


Figure 5.3: Signal variations during the execution of the “knock” gesture.

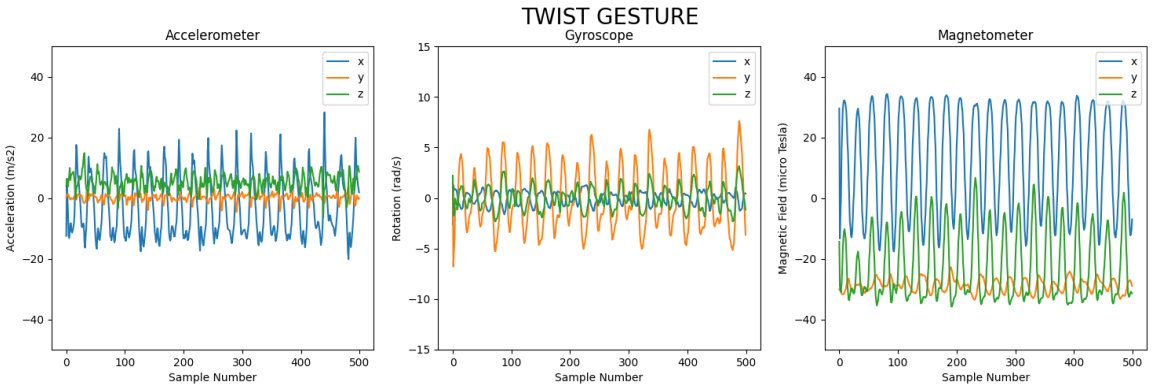


Figure 5.4: Signal variations during the execution of the wrist “twist” gesture.

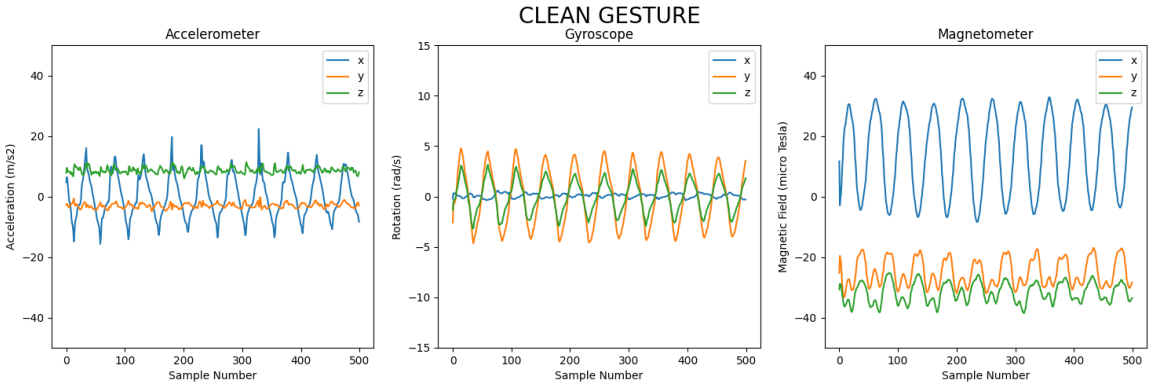


Figure 5.5: Signal variations during the execution of the “clean” gesture.

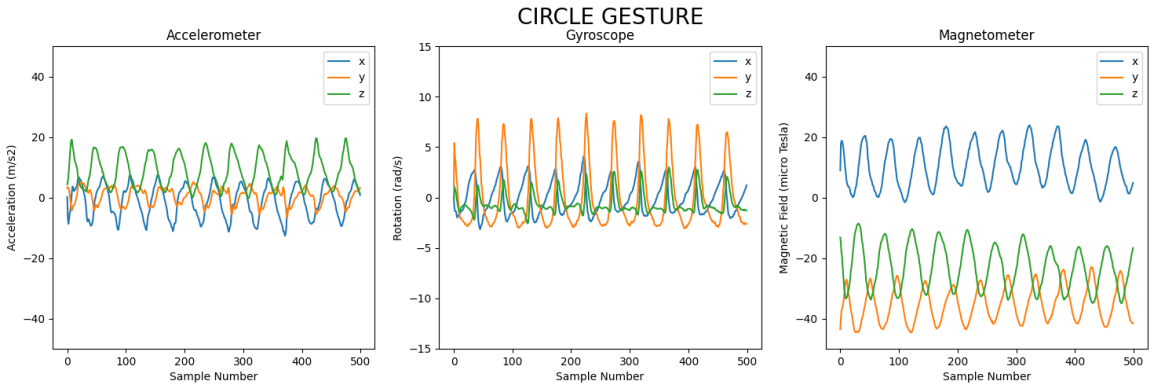


Figure 5.6: Signal variations during the execution of the "circle" gesture.

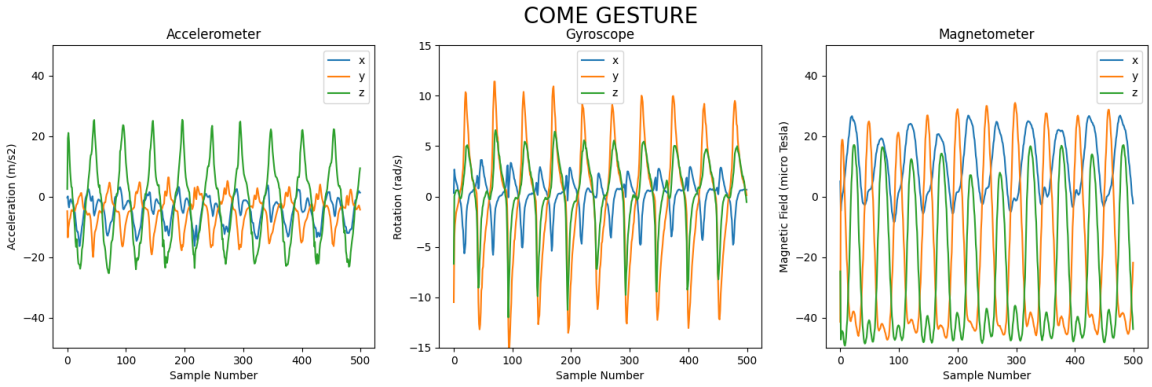


Figure 5.7: Signal variations during the execution of the "come" gesture.

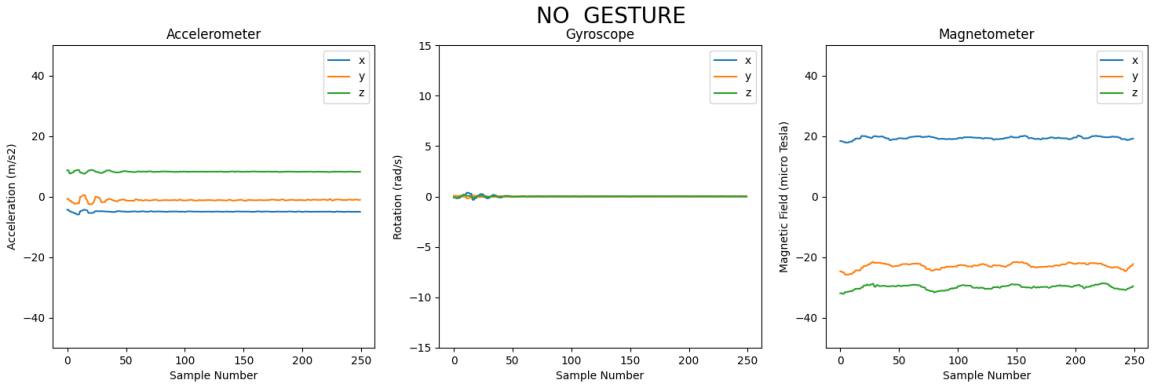


Figure 5.8: Signal variations during the execution of the "no gesture".

These plots depict how the various signals change during the execution of different gestures. Based on a naked-eye analysis, there are some clear patterns in the signals generated, but the differences in amplitude and speeds between different executions, as well as the possibility of distortion added by unintentional shakes, must be taken into account, especially given the context of the problem and the population addressed.

Threshold-based solutions were successfully used in early work involving primarily activity recognition to distinguish between different sets of activities, but with the introduction of dynamic arm movements utilizing a single smartwatch, other approaches based on machine learning and deep learning began to be experimented due to improvements in those topics as well as due to the increasing capabilities of the devices capturing and receiving the signals.

5.2.3 Classification Approach and Results

5.2.3.1 Feature Selection

In this experiment, the **RelieF** algorithm for feature selection was used. RelieF is an algorithm, initially proposed for binary classification problems, that uses a filter method approach to do feature selection. RelieF gives a score for each feature based on differences between the nearest neighbor instance pairs, and since its creation, a few RelieF-based feature selection algorithms have been created to tackle problems that the original one did not, such as multi-class classification.

In this study, the **TuRF** variant was chosen. **TuRF** stands for Tuned Relief and is especially used to address noise in large feature spaces, being thus relevant in the context of the problem.

The 20 most relevant features selected using **TuRF** are presented in Table 5.3, and were the features chosen for classification in the later stages, instead of the extraction of all 84 time-domain features.

Sensor	Statistical Measure
Accelerometer	r_{xy}, r_{xz}, r_{yz}, σ_x, σ_y, σ_z, range_z
Gyroscope	r_{xy}, r_{xz}, r_{yz}, σ_y, range_y
Magnetometer	r_{xy}, r_{xz}, r_{yz}, σ_y, σ_z, range_x, range_y, range_z

Table 5.3: Top 20 features selected from the sensor data from the first experiment.

5.2.3.2 Classifiers and Evaluation Approach

In this initial evaluation, taking into account the limited size of our dataset, the following classic classifiers were contemplated: support vector machines (SVM), decision tree (DT), random forest (RF), and Gaussian Naïve Bayes (GNB). Python's "scikit-learn" library (version 0.24.2) [68] was used for model training and testing. The default values were used for the classifiers' hyperparameters, except the SVM's kernel (linear kernel), due to the fact that for this experiment the linear kernel held better results when compared to the default kernel (RBF).

Two evaluation approaches were followed. Initially, the performance of the model obtained with each classifier was evaluated using 10-fold cross-validation over the data from participant one. Secondly, the classifier that yielded the best results from the first approach was used to train a model with all data from participant one, which was then tested with data from never-seen subjects (participants two and three). This last approach was used to verify how

good was the generalization of a model trained with data from a group of subjects and used to predict gestures based on data from never-seen subjects, which is very relevant for a real-world scenario.

Given the classes were balanced (i.e., the same number of instances for each gesture or class), the metric chosen for evaluation was the accuracy.

5.2.3.3 Results

Table 5.4 presents the results achieved using the classifiers mentioned, as well as data from Participant 1 for both training and testing.

Gesture	**SVM**	**DT**	**RF**	**GNB**
ALL	99.7	98.6	99.7	99.6
Knock	100.0	99.7	100.0	100.0
Twist	99.3	98.1	99.8	99.4
Clean	99.3	96.1	99.3	99.8
Circle	100.0	98.2	99.9	99.1
Come	99.9	97.8	100.0	100.0
No Gesture	99.8	99.3	99.7	99.7

Table 5.4: Mean accuracy results (%) for the 4 classifiers, using Participant 1's data for training and testing (10-fold cross-validation).

From the results is observable that all classifiers yielded great results with the overall accuracy being higher than 98%. Furthermore, all gestures were recognized with similar accuracy (>99%) for all classifiers except the DT (presenting an accuracy of at least 96%).

The worst result was obtained for the "Clean" gesture using DT (accuracy of 96%) and overall the "Clean" gesture presents the worst results among the different classifiers. However, the results for the classification of that gesture are still very high considering it got a minimum of 99% except for, as stated, DT.

The best model performance was achieved with the SVM and RF algorithms, with similar overall accuracy but with varying accuracy values for the different gestures. While SVM was capable of delivering perfect accuracy values for the "Knock" and "Circle" gestures, RF presented perfect recognition for the "Knock" and "Come" gestures. Nevertheless, the remaining gestures presented accuracy results over 99%, which is excellent.

The results for classification when using the model trained with SVM and all data from participant 1 are presented in Table 5.5. The table only shows the results using SVM, since it leads to the best results on this second approach when compared with RF.

When considering the performance of a model tested with data from never-seen subjects, it is expected that the performance decreases comparatively to a subject-dependent solution, which is exactly what occurs and is observable, with an overall accuracy of 94% for all gestures and subjects, which is considerably lower than the results shown in Table 5.4 that contemplated only data from participant one for both training and testing.

When considering the accuracy results for each individual gesture, the highest accuracy is achieved for the "Knock" and "Come" gestures, while for the remaining gestures the accuracy varied between 81% and 96% with the "Clean" gesture holding, once again, the worsts

Participant	ALL	Knock	Twist	Clean	Circle	Come
2	93.7	99.5	100.0	69.2	100.0	100.0
3	93.4	100.0	78.2	100.0	88.7	100.0
Both	93.6	99.7	91.7	80.9	95.7	100.0

Table 5.5: Accuracy results (%) obtained when training a model with SVM and data from Participant 1, and testing it with data from Participants 2 and 3.

result. It is also interesting to note that, with the exception of the "Come" gesture, which held perfect accuracy results for both participants, for "Twist", "Clean" and "Circle", despite the fact that for one of the participants' classification was flawless, the accuracy was lower for the other participant. Regardless, both individuals' results are similar and plenty good, with an accuracy between 93% and 94% considering the mean for all gestures.

5.2.3.4 Conclusion

The first preliminary results obtained in this experiment for assessing the performance of machine learning algorithms to recognize a six-class set of gestures, using only one wrist-mounted wearable, are very promising. Although data are scarce and one of the drawbacks of this experiment, it was important to assess that, first, a subject-dependent solution is viable, and second, that a subject-independent solution seems to be viable with the results suggesting that a model trained from a small set of participants may result in a model with good generalization, which is very useful in real-world scenarios.

The results are reassuring and in line with the performance of related systems (e.g., [44]), but, nevertheless, the gathering of data from more participants, with more gesture instances and possibly different gestures, is required to obtain more convincing results. Also, with more data, the performance of different models with different sliding window sizes, window overlaps, and feature selectors may be evaluated to understand which combinations yield the best results for gesture recognition in the bed scenario.

5.3 Experiment with Wearable

Following the initial experiment, which yielded promising results for gesture recognition in a bed scenario using only one wearable, a more detailed experiment was conducted to get more definitive results on the usage of gestures in the bed setting.

Similar to the first experiment, a study on the effect of the classifier on the model's performance was investigated using classic machine learning algorithms and considering a subject-dependent scenario. Another experiment was also carried out using a leave-one-subject-out cross-validation approach to study subject independence. On the other hand, contrary to the first experiment, in this one, due to having more data and therefore more instances to train and test, an additional study on the effect of different sliding window sizes, window overlaps, and features selected, was done.

All findings were analyzed and discussed to determine which classification algorithm,

feature set, sliding window size, and window overlap combination is optimum for gesture recognition in a bed situation for people with aphasia or anyone with speech impairment.

5.3.1 Experimental Setup and Protocol

The first step of this experiment was to gather data from more participants. For that, a convenience sample of 10 able-bodied adults (5 male, 5 female) were recruited among friends and family. The characteristics of the sample can be found in Table 5.6. Similar to the first experiment, all participants involved in this second experiment were not affected by aphasia or any other liability to the upper body.

No. users	Sex (M/F)	Age	Weight	Height	Right/Left handed
10	5/5	44.3 [22, 76]	77.9 [58,91]	176.5 [164, 185]	10/0

Table 5.6: User characterization for the participants involved in the second experiment. The values for age, weight, and height are presented as mean [minimum, maximum].

Figure 5.9: Participant laying in bed ready to start the data gathering procedure.

The data gathering for this experiment was conducted over different days depending on the availability of the participants, however, the recording for a participant was always made during the same day, and the process took a maximum of thirty minutes. The bed and equipment for recording were the same throughout all recordings, and two special applications for the recording of data were developed to ease this process (application described in Appendix A).

All participants in the gathering of data for this experiment signed a consent where the experiment was described, along with the conditions and data to be gathered during the executions. After agreeing and signing the consent, they were given a sequential identifier with three figures that identified the participants providing anonymity. They were then asked their age, height, weight, and dominant arm for statistical purposes, and before beginning the trial they were free to ask any question. There was not a single participant that showed signs of doubt or any other relevant worry.

Participants were then asked to place the wearable device tightly on their dominant wrist, the same they indicated before. This wearable device, contrary to the first experiment, was a Wear OS smartwatch, containing a 3D accelerometer, gyroscope, and magnetometer. With the wearable device correctly placed on their dominant wrist, they were instructed on which gestures to perform, as well as the protocol. Each gesture was given a name and a brief explanation of the movement to be performed. The protocol stipulated that each gesture should be performed 10 times, for 5 continuous seconds each, in different sessions and in a random order to eliminate bias that could result from performing 10 repetitions of the same gesture one after another or different gestures but in the same order. The protocol

also stipulated that before beginning the execution of a gesture, the person in charge of commanding the recording would indicate the name of the gesture and the moment to start and stop the execution. This was done to help the process of recording without any trouble.

Participants were encouraged to perform every gesture in the most natural way possible with the speed and amplitude they intended and found the most natural. They were also informed that, at any moment, they could refuse to continue the trial, leaving at any moment they felt it was enough. Nevertheless, that did not happen and every participant concluded the process successfully and with no trouble.

If no question was brought on the protocol or gestures to be executed, the participants were asked to lay in bed, with arms parallel to the body, as demonstrated in Figure 5.9. In this position, they were asked if there was any question or worry and were once again encouraged to execute the gestures the most naturally possible.

5.3.2 Dataset Characterization

Each participant provided ten recordings of five seconds for each of the gestures. The recordings for each participant were saved in a folder named with the identifier of a participant with folders for each gesture inside it.

Data were recorded at a sampling rate of 50 hertz, meaning that for every twenty milliseconds a new value for each of the sensors was recorded and saved, leading to two hundred and 50 data points[2] for each recording of a gesture, totaling 2,500 for 10 recordings of each gesture and a total of 15,000 for one participant.

Table 5.7 presents an overview of the collected data. The collected data were later processed to extract important features to allow classification. To provide a good in-depth analysis, different windows (either 1 or 2 seconds) and different window overlaps (0%, 50%, and 96 %) were used when extracting the important features, leading to six different datasets with the different possible combinations of sliding window size and window overlap. These datasets were saved in a specific Comma-separated values (CSV) file, named after each of the variables (*e.g.,* **participant001_2secWindow_96overlap.csv**, stands for a dataset from the participant with the identifier '001' when considering a 2-second sliding window size and 96% window overlap).

Sliding-Window (s)	1			2		
Overlap (%)	0	50	96	0	50	96
No. Instances Per gesture type	500	900	10,100	400	700	7,600
Total	3,000	5,400	60,600	2,400	4,200	45,600

Table 5.7: Total number of instances obtained in the second experiment, for each combination of sliding window size and overlap.

The plots presented in Fig. 5.10, 5.11, 5.12, 5.13, 5.14, 5.15 below illustrate how the various signals provided by the smartwatch change during the execution of each gesture. Similar to the plots produced in the first experiment (with a smartphone), there are some clear indications that some features may be significant to discern between different gestures.

[2]Data points include a value for every axis of the accelerometer, gyroscope, and magnetometer, and a timestamp.

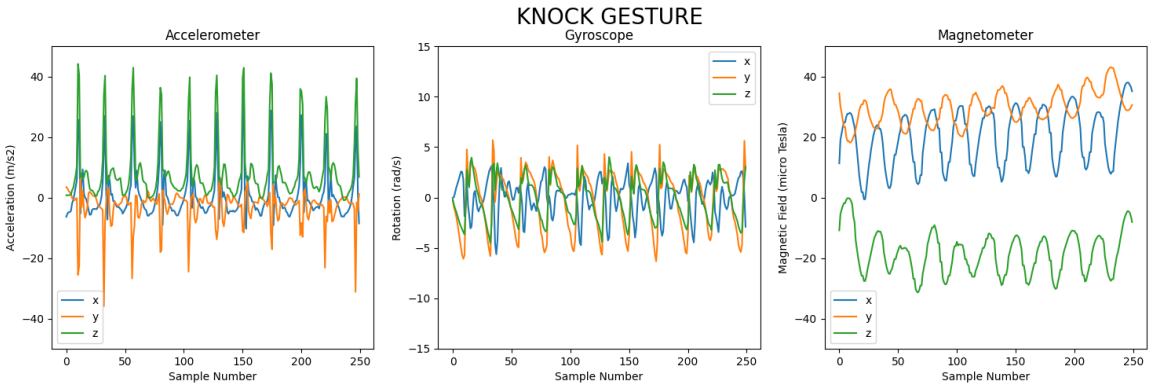


Figure 5.10: Signal variations during the execution of the wrist "knock" gesture.

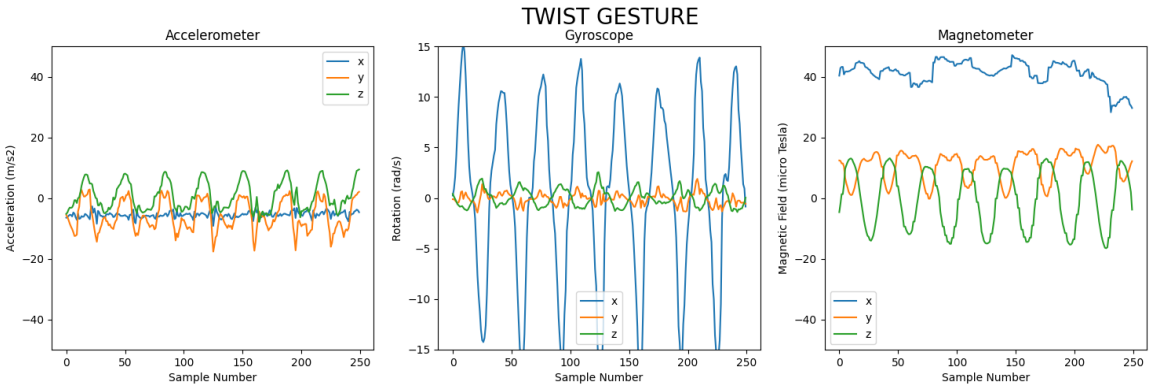


Figure 5.11: Signal variations during the execution of the wrist "twist" gesture.

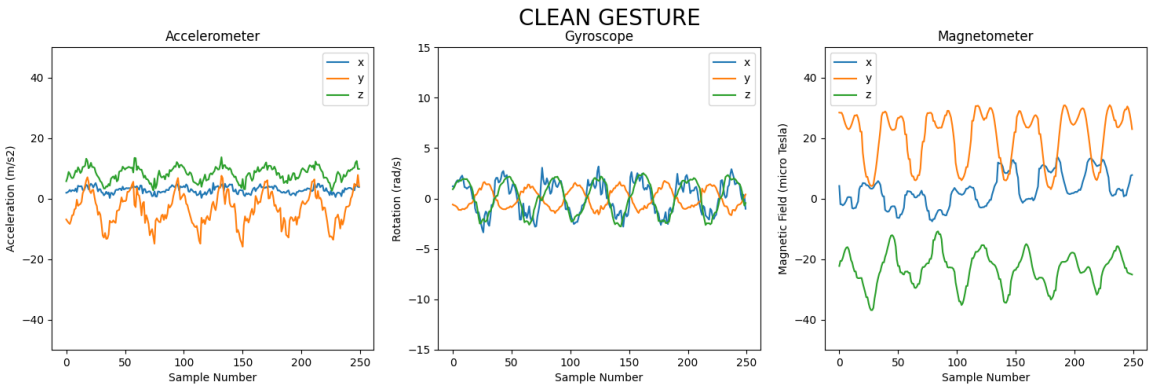


Figure 5.12: Signal variations during the execution of the "clean" gesture.

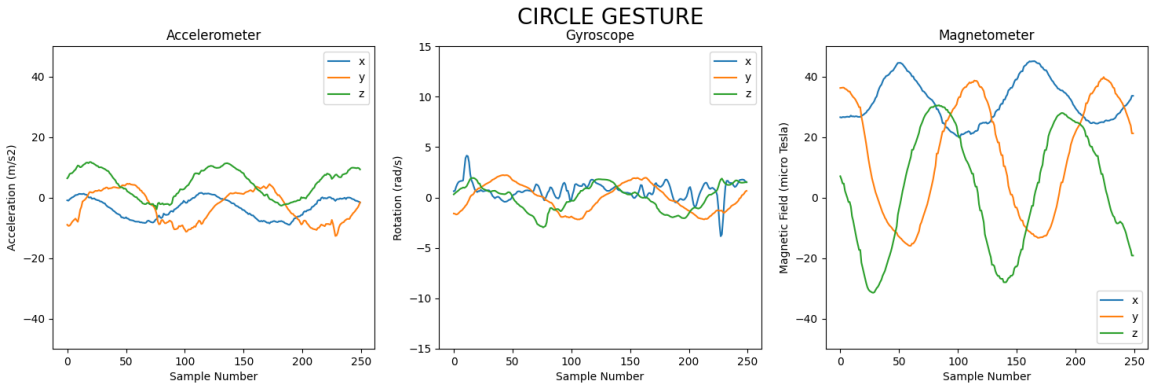


Figure 5.13: Signal variations during the execution of the “circle” gesture.

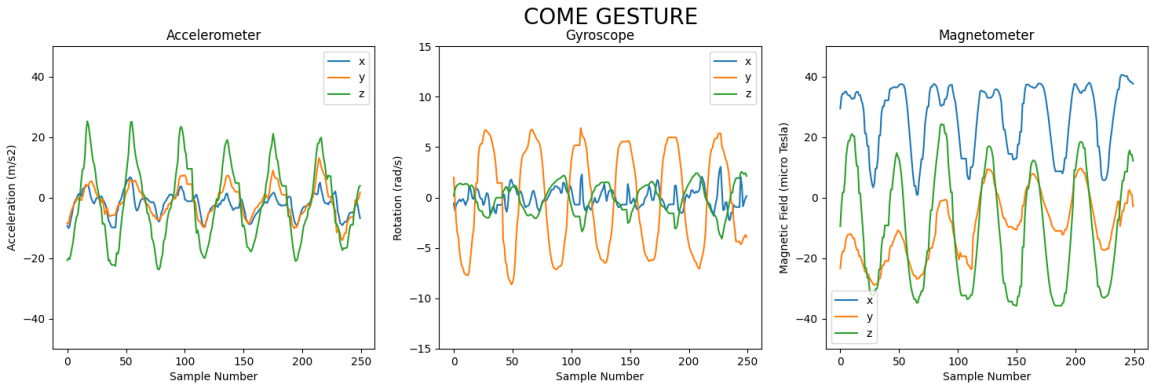


Figure 5.14: Signal variations during the execution of the “come” gesture.

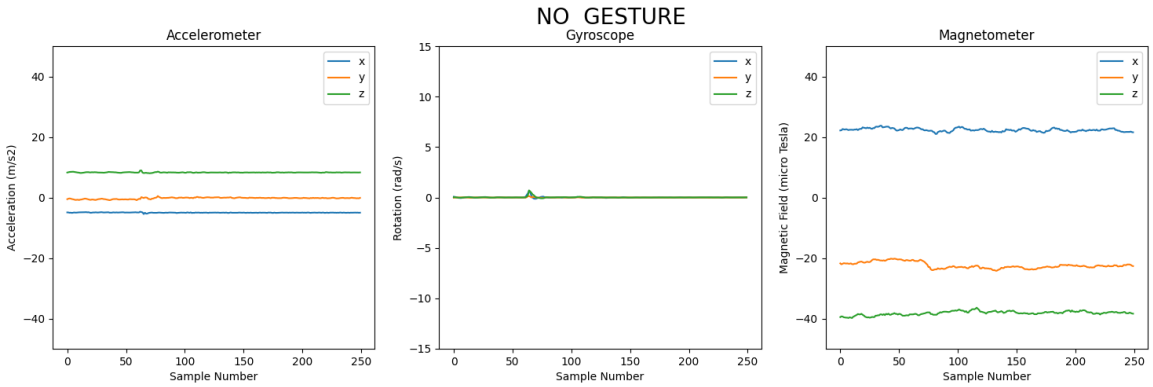


Figure 5.15: Signal variations during the execution of the “no gesture”.

Although the plots obtained are comparable to the ones obtained on the first experiment, they are generally in different ranges, yet the patterns produced are similar . This suggests that, while the sensors from the smartphone used in the first experiment, and the smartwatch used in the second are comparable, they may have different resolutions. However, because

the equipment used did not allow to access the specifications of each sensor, no conclusions could be drawn in this respect.

Different sliding windows and overlaps will present different results depending on the classifier used. Even though previous research on gesture recognition came up with different optimal combinations of overlaps and sliding windows, due to the unique scenario and environment involved in this dissertation, the following subsections present an investigation of the different combinations of feature selectors, classifiers, sliding window sizes, and window overlaps.

5.3.3 Evaluation Approach and Results

The data recorded with the contribution of ten participants led to different data sets with the particular number of instances presented in Table 5.7. In the next topics, the effect of different classifiers, different sliding window sizes, and different window overlaps will be addressed in two distinct scenarios: subject-dependent and independent. For the subject dependent case, considering a model is trained and tested with data for each participant separately (using a 10-fold cross-validation approach), and a subject-independence scenario, that leaves, each time, one user out of the training process to assess the performance of a model trained with data from the rest of the participants on that users' data (leave-one-subject-out cross-validation approach).

Additionally, an evaluation of the use of two feature selectors will be conducted and compared to the feature selection process done in the first experiment. For this evaluation, two Feature Selectors will be used and evaluated for the different data sets obtained from different combinations of window overlaps and sliding window sizes. This evaluation intends to provide an overview of which feature selector holds better results as well as assess the minimum number of features needed to obtain a good classification result.

Since our dataset is balanced, the following evaluation metrics were considered: overall accuracy (5.1) and F1 score (5.2). In our system, it is important to avoid false negatives when considering "No gesture" as the positive class, i.e., avoid detecting a gesture when there is no gesture. Therefore, the false-negative rate (FNR) for the "No gesture" class was also computed using (5.6).

$$\text{Overall accuracy } (\%) = \frac{TP + TN}{TP + TN + FP + FN} \times 100 \tag{5.1}$$

$$\text{Overall F1 score } (\%) = \frac{\sum_{i=1}^{C} \text{F1}(c_i)}{C} \tag{5.2}$$

$$\text{Class F1 score } (\%) = 2 \times \frac{\text{class precision} \times \text{class recall}}{\text{class precision} + \text{class recall}} \tag{5.3}$$

$$\text{Class precision } (\%) = \frac{TP}{TP + FP} \times 100 \tag{5.4}$$

$$\text{Class recall } (\%) = \frac{TP}{TP + FN} \times 100 \tag{5.5}$$

$$\text{Class FNR } (\%) = \frac{FN}{TP + FN} \times 100 = 1 - \text{class recall} \tag{5.6}$$

In (5.4), (5.5) and (5.6), TP, TN, FP and FN correspond to:

- True positives (TP): number of instances correctly classified as belonging to the considered class;
- True negatives (TN): number of instances correctly classified as belonging to a class other than the one considered;
- False positives (FP): number of instances incorrectly classified as belonging to the considered class;
- False negatives (FN): number of instances incorrectly classified as belonging to a class other than the one considered.

In (5.1), TP, TN, FP and FN correspond to the sum of the corresponding values considering all classes. In (5.2), C is the number of classes and F1(c_i) is the F1 score for class c_i.

5.3.3.1 Feature Selection

The challenge of feature selection is to decide what feature set can be used for classification without compromising the accuracy of classification or requiring a large computational power when using a large set of features.

The same process of feature extraction from the first experiment was used in this second one, but this time it was performed for different combinations of window sizes and overlaps. For each sliding window, 84 time-domain features were extracted, which were chosen over frequency domain features due to being less computationally expensive. These features are presented in Table 4.3 and are mostly based on past work done in gesture and activity recognition [46]. The features are presented in Table 4.3.

To evaluate the performance of two feature selectors, a similar technique to the Sequential Forward Selection (SFS) [69] was used. The two filter-based feature selectors selected chosen were ANOVA F-Value and, similarly to the first experiment, RelieF. The process of evaluation is divided into two distinct stages:

1. **Feature Ranking:** each feature selector ranks the features from 1 to 84, for each dataset corresponding to each combination of sliding window size and overlap.
2. **Accuracy for N Features:** for each combination, each classifier is assessed using the features selected in the first step, starting with the most prominent feature and gradually increasing one feature at a time. The accuracy of each additional feature is logged to determine when adding new features no longer benefits the model being assessed.

5.3.3.1.1 Feature Ranking

The results obtained for the feature ranking of each combination of window size and overlap are very extensive and for this reason, are not presented, but a few observations may be made.

With RelieF, regardless of the window size and overlap, the features' ranking is similar. For instance, considering the top twenty features for each combination, only the cases of the range of the gyroscope's y-axis, the standard deviation of the magnetometer's y-axis, and the standard deviation of the magnetometer's x-axis, occur once, with the rest of the features being selected in at least three combinations, with most of them being selected every time. It is also interesting to note that RelieF tends to prefer features related to the correlation between pairs of axes of the signal axis, with standard deviations and variance of the signals coming further down on the list of preferences.

Using ANOVA F-Value, although the rankings of features vary more between different combinations of overlaps and windows, the features among different combinations remain the same. Although less noticeable due to the slightly more varied feature scoring, the correlations of the signal axis, standard deviations, and range remain on the top of the features selected. Similar to what was done for RelieF, if considering the top twenty features selected for each combination, only the cases of the mean of the magnetometer z-axis, integral of the magnetometer z-axis, and the standard deviation of the magnetometer, x-axis, are not present in all combinations. Furthermore, similarly to what occurred with RelieF, the feature sum of all squares gets left on the bottom of the rankings in all cases.

When comparing both feature selectors, it gets clear that both select features using different criteria, nevertheless they end up selecting a portion of the same features among them with eleven of the top twenty features being selected by both feature selectors. Another interesting aspect is that RelieF, compared to ANOVA F-Value, selects more features calculated over the values of the magnetometer sensor, with, for instance, the example of ANOVA F-Values, using a 96% overlap and a 1-second sliding window, selecting four features related to the magnetometer which is almost three times less of the RelieF algorithm that selects eleven features calculated over data from the magnetometer.

When taking the results of feature selection from the first experiment into consideration, what we can observe is that, although with different data sets, the similarity's between the features selected are astonishing.

The features selected for the first experiment are present in Table 4.3 and, if compared with a comparable set of twenty features for each feature selector of this second experiment, what we can observe is that the correlations between the signal axis are the most predominately selected features, appearing in all cases independently of the feature selector or data set chosen, and goes in line with what was experienced by Tchuente et al. work [46]. Another important aspect is that all other features that are similar between them are the range of all axis of the magnetometer and the standard variation of the y-axis and z-axis of the magnetometer also. This is a possible indication that the magnetometer is an important sensor to have in account when considering data from never seen subjects, but further investigation needs to be conducted to assess it.

5.3.3.1.2 Accuracy for N Features

With the stage of Feature Ranking completed, the accuracy for each classifier using an increasing number of features was evaluated. The five different classifiers (SVM, DT, RF, kNN, and GB) considered for this second experiment were evaluated to determine the minimum

number of features required to achieve an acceptable classification result.

In this experiment, each classifier was trained and tested over datasets containing data from all users combined, with the same window size and overlap. These classifiers were used to train and test a model with an increasing number of features and the mean accuracy was obtained and assessed using 10-fold cross-validation to verify when the gain in accuracy ceased being noticeable.

The outcomes of the experiment described are once again remarkable. Regarding RelieF feature selection, the accuracy improves, regardless of the combination of classifiers, overlaps, and windows, with an increasing number of features until it reaches the mark of ten to twenty features. On the other hand, ANOVA F-Value appears to be more unstable in that regard, with some combinations requiring close to thirty features to stabilize the accuracy, while others do not, with only the need of ten to fifteen features. This is important to notice especially if low computational power is a requirement of the system, allowing less processing power to obtain a similar level of accuracy. One other important aspect to notice is that, independently of the feature selector, DT produces the lowest mean accuracy values even with a high number of features selected.

To define a heuristic that helps find the minimum number of features for which the accuracy stops increasing by a given amount, it was decided that the number of features chosen would be the first to obtain an accuracy value of at least 1%/2%/3% less than the maximum for that combination. For instance, considering a criterion of 2%, RelieF and a window of 1 s with 0% overlap, the maximum accuracy obtained is 95%, but 13 features are enough to get 94% accuracy, which goes accordingly to the heuristic defined.

A summary of the information gathered can be observed in Table 5.8, presenting the minimum, maximum, mean, median, and mode, number of features with three different criterion values (1%, 2%, and 3%). The values for each criterion are obtained as a mean of the different combinations.

Feature Selector	**Criterion(%)**	**Minimum**	**Maximum**	**Mean**	**Median**	**Mode**
RelieF	1	7	79	27.2	21.0	18.0
	2	7	43	16.1	15.5	13.0
	3	5	24	12.2	11.0	11.0
ANOVA F-Value	1	7	79	28.4	27.5	8.0
	2	6	43	15.6	11.5	7.0
	3	5	27	10.3	8.0	7.0

Table 5.8: Minimum, maximum, mean, median, and mode, number of features needed with three different criterion values (1%, 2%, and 3%)

The number of features is closely the same when comparing both feature selectors with the same stopping criteria. The fact that the median is lower than the mean is an indication that there is some combination of sliding window and overlap that demands more features, making the median a better indicator. An interesting fact is that the mode of the ANOVA F-Value has a lower mode compared to RelieF, indicating that there is a majority of cases where a lower number of features is can be used.

To conclude, although some variations in the number of features required to reach a stable number of features may occur, to compare both feature selection approaches, the same number

of features for both will be selected, and the median using a 2% stopping criteria was chosen, due to presenting a good compromise between the number of features and accuracy. With that said, since Relief has a median of 15.5% and ANOVA F-Value has a median of 11.5%, selecting the 16 top features seems a reasonable number of features to consider.

Lastly, while the results seem to be conclusive that the features selected by both feature selectors are good regardless of the sliding window size, overlap chosen, the results for subject-dependent and subject-independent being described next will use the features already selected previously in the first experiment. This is done, first, to withdraw conclusions on a feature set chosen from data that is going going to be used in later stages to train and test models, and also because the feature set seems to be rather similar, making it a good candidate to be used at this stage.

5.3.3.2 Subject Dependence

As indicated in the introduction of this section, the next step is to verify the effect of the different classifiers, window sizes, and window overlaps for the subject-dependent case. The results of this experiment will help understand the effects of the different variables, as well as obtain the combination yielding the best results for classification of movements.

For this experiment, five classic machine learning classifiers, one more than what was used in the first experiment, were assessed: **support vector machines** (SVM), **decision tree** (DT), **Random Forest** (RF), and **Gaussian Naïve Bayes** (GNB). This time **K-Nearest-Neighbor** (kNN) is also being considered for a more in-depth analysis of classic machine learning techniques.

Python's "scikit-learn" library (version 0.24.2) [68] was used for model training and testing. The default values were used for the classifiers hyperparameters except for the SVM kernel since the results of the first experiment using a Linear Kernel held better results. Because these classifiers are often used in machine learning research, they provide a useful starting point for understanding each one's behavior in the bed scenario described in this dissertation.

The performance of the models obtained with each combination of classifier, window size, and window overlap, were evaluated over the balanced datasets from every participant individually, using 10-fold cross-validation. This procedure produces, for each participant, 30 combinations of classifier, window overlap, and sliding window size, for a total of 300 for all participants.

For the obtained results, three important effects need to be discussed: the classifier, the window overlap, and the sliding window size effect. Following that, the impact of each variable will be explored, with figures to assist the understanding of the variations.

5.3.3.2.1 Classifier effect

Five different classifiers were used over different combinations of features, overlaps, and windows. To find which classifier yields the best results for gesture recognition in the specific scenario described in this dissertation, the mean for the accuracy, precision, recall, and F1-score were calculated over a 10-fold cross-validation test.

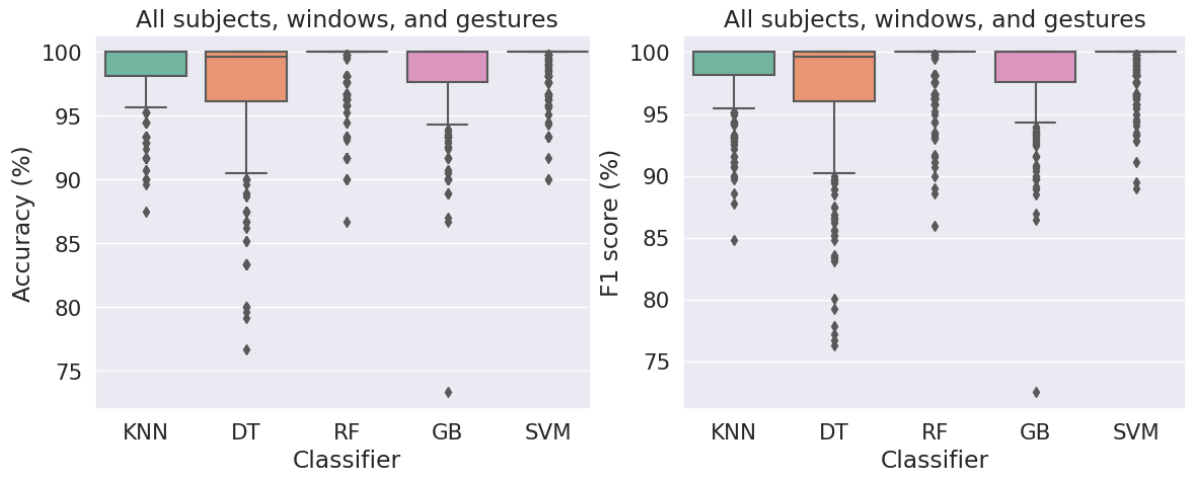


Figure 5.16: Boxplot for the Mean accuracy and F1-Score results for the classifier effect on a subject-dependent scenario.

Figure 5.16 presents the box plots of the values of the accuracy's and F1-score for each fold and classifier, independently of window, overlap, and user.

When considering the effect of the different classifiers it is observable that SVM and RF hold the better results, while DT presents the worst. Both SVM and RF present almost no variability, and median values close to 100%, with SVM presenting fewer outliers, compared to RF, in both the mean accuracy and F1-score values. It is also observable that the GB and kNN classifiers yield closely the same results, with most accuracy values over 95%, but not close to the ones provided by RF and SVM. Nevertheless, the median values presented by GB and kNN are still high but due to the variability of the Mean accuracy and F1-Score SVM and RF are considered better classifiers for the scenario being addressed.

Table 5.9 presents the mean value for the overall accuracy and F1-score over the 10 cross-validation folds resulting from the five classifiers over each user individually. The comparison is done between users to allow a more in-depth analysis of the performance of the classifiers over different users.

	DT		GB		KNN		RF		SVM	
Participant Identifier	**Mean accuracy(%)**	**Mean F1(%)**	**Mean accuracy(%)**	**Mean F1(%)**	**Mean accuracy(%)**	**Mean F1(%)**	**Mean accuracy(%)**	**Mean F1(%)**	**Mean accuracy(%)**	**Mean F1(%)**
1	97.2	97.2	98.7	98.7	98.3	97.9	98.7	98.7	99.5	99.5
2	98.7	98.7	99.5	99.5	99.1	99.0	99.9	99.9	99.9	99.9
3	93.6	93.3	94.7	94.6	97.4	97.3	98.1	98.1	98.9	98.9
4	99.2	99.2	99.4	99.4	99.5	99.4	99.9	99.9	99.8	99.8
5	97.2	97.2	99.5	99.5	99.1	99.0	99.9	99.9	100.0	100.0
6	98.6	98.6	99.8	99.8	99.4	99.4	99.7	99.7	99.9	99.9
7	96.6	96.6	97.9	97.8	98.8	98.7	99.2	99.1	98.5	98.5
8	96.5	96.4	96.8	96.8	98.1	98.0	98.6	98.5	98.4	98.3
9	98.1	98.0	98.8	98.8	99.2	99.1	99.1	99.1	99.8	99.8
10	98.4	98.4	99.7	99.7	99.9	99.9	99.9	99.9	99.9	99.9

Table 5.9: Classifier effect over each participant on a subject dependence scenario.

What can be taken from the results is that:

1. DT, as expected, presents the worst results for classification, yet, for most of the users

it still prevails with an accuracy over 93%, which is still low compared to the other classifiers, but high concerning the classification of movements.

2. SVM presents the best results in the majority of the participants, showing good results even in the toughest participants where other classifiers show worse results. kNN and RF come after with also great results but with instances where it fails more than compared to SVM.

3. The third participant has the worst results, with DT failing the most to recognize the movements (accuracy of around 94%). SVM and RF models, on the other hand, show versatility by maintaining good accuracy overall, even in the most difficult cases like participant 3.

4. Although kNN and GB in most cases present a good classification result, close to the ones provided by SVM and RF, it fails to provide the same level of accuracy in other scenarios (*e.g.,* participants 3 and 8) where the latest shows consistently good values.

5.3.3.2.2 Window Size and Overlap Effect

When considering alternative sliding window sizes, the main drawback with its increase is that it takes longer to recognize a specific movement and therefore delays the response of any system that reacts upon a classification. This may not be a problem in situations where a quick response of the system is not required, but it could be a problem in other situations. Due to the scenario in which these gesture recognition models will be employed, it is critical to have a quick reaction. For that reason, only two sliding window sizes were considered: 1 and 2 seconds. Other sliding window sizes could be considered, especially those in between both, or even intervals of less than a second, but it was decided that, for the first experiment with different sliding window sizes, those two would be enough.

When taking into account the overlap used, there are two main focal points to consider: using a bigger overlap results in a greater amount of data that takes longer to process, and, even though not being considered, for now, using a bigger overlap may lead to overfitting of the models, resulting in worse classification results when considering data from never seen subjects.

Figures 5.17 and 5.18 present the boxplot for the mean 10-fold cross-validation accuracy and F1-score results with the use of different window sizes and window overlaps over the different tested classifiers.

Regarding the sliding window size only, what can be taken from the results is that the 2-second sliding window size provides better results with a small margin over the 1-second sliding window, especially when considering accuracy results for SVM and RF. The remaining classifiers improve on the accuracy and F1-score but are not as evident.

Nonetheless, the use of either sliding window size seems viable independently of the classifier chosen. Although some cases present better results with one over the other, the differences are small and maybe disregarded if the system where the models are being made for has the requirement of a higher sliding window size.

Regarding only the overlap chosen, the results show that increasing the overlap improves the accuracy and F1-Score for all classifiers.

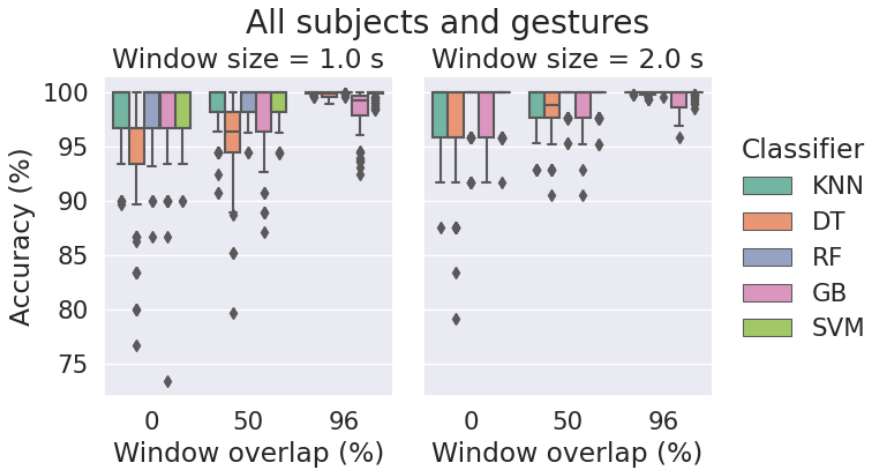


Figure 5.17: Boxplot for the Mean accuracy values considering the overlap and window effect over each different classifier, for a subject-dependent scenario.

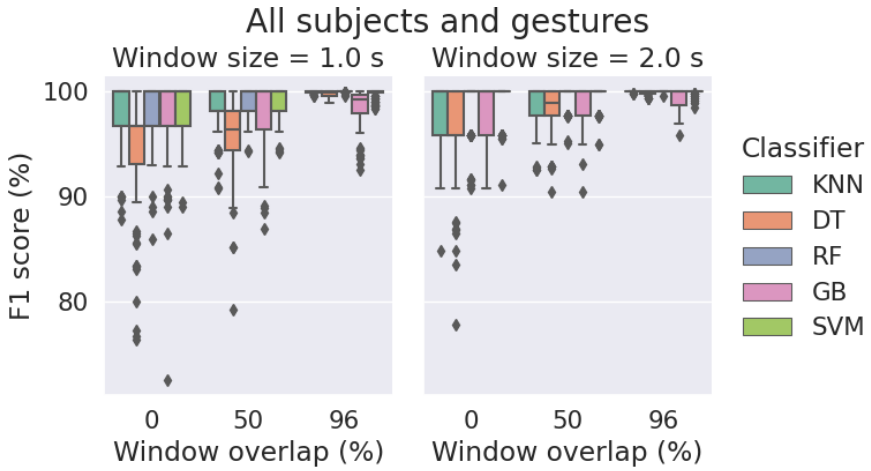


Figure 5.18: Boxplot for the Mean F1-Score values considering the overlap and window effect over each different classifiers, for a subject-dependent scenario.

Figures 5.19 and 5.20 present the accuracy and F1-score mean results considering SVM and RF individually since they were the two best resulting classifiers.

Once again, regarding the sliding window size and overlap chosen, the results are similar to the overall results. If using either SVM or RF, as stated before, the 2-second window yields better results even with an overlap of 0%. Nonetheless, if using a 96% overlap, the results with a 1-second sliding window size are also perfect, making both approaches viable.

5.3.3.2.3 Results per gesture

In the above topics, conclusions were taken on the effect of multiple variables individually. It was found that SVM and RF produce, on average, and considering only the effect of the classifier, the best results for classification of the movements recorded. However, due to

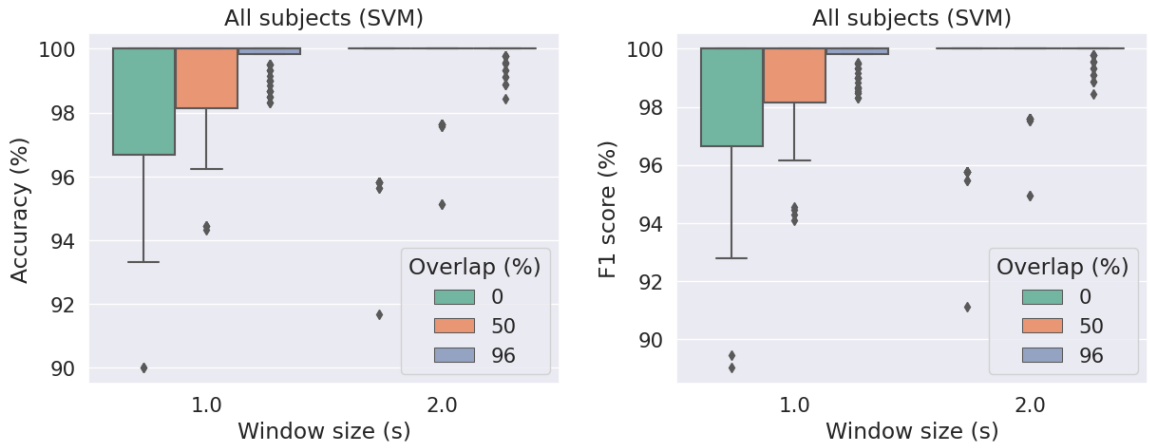


Figure 5.19: Boxplot for the Mean Accuracy and F1-Score results for different combinations of sliding window size and overlap, considering SVM only.

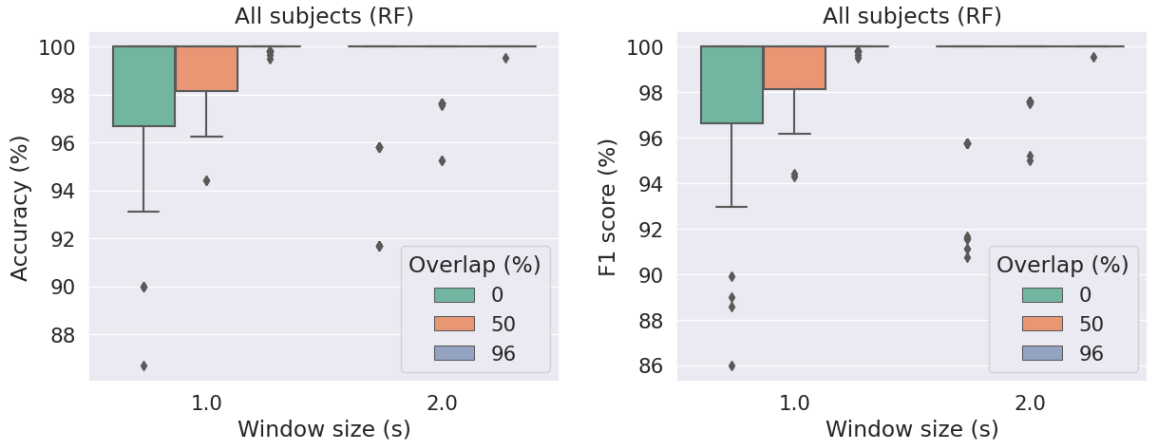


Figure 5.20: Boxplot for the Mean Accuracy and F1-Score results for different combinations of sliding window size and overlap, considering only RF.

---

presenting lesser outliers compared to SVM, RF will be preferred.

It was also revealed that, in a subject-dependent scenario, the overlap choice entails, on average, remarkably better results when using higher overlaps. It was also revealed that the sliding window size of choice leads to improvements when choosing the 2-second one over the 1-second sliding window. Nevertheless, if considering only the two best classifiers, SVM and RF, a 1-second sliding window with 96% overlap also shows great results, comparable to those obtained with a 2-second sliding window.

To understand how well each gesture is classified, the results for each participant using the overall best combination of variables - RF, 2-second sliding window size, and 96% window overlap - were taken and the results are presented in Table 5.10. The metric chosen to evaluate the recognition for each gesture was the F1-Value that takes the value from both precision and recall, but the overall accuracy, F1-Score, and FNR (for the No-gesture) for each participant are also presented.

The results obtained are similar, presenting flawless F1-Score for all participants, except

Participant Identifier	Knock F1(%)	Twist F1(%)	Clean F1(%)	Circle F1(%)	Come F1(%)	No gesture F1(%)	Accuracy(%)	F1-Score(%)	FNR (no gesture)(%)
1	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	0.0
2	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	0.0
3	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	0.0
4	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	0.0
5	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	0.0
6	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	0.0
7	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	0.0
8	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	0.0
9	100.0	100.0	99.9	100.0	100.0	99.9	99.9	99.9	0.0
10	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	0.0
Total	**100.0**	**100.0**	**100.0**	**100**	**100.0**	**100.0**	**100.0**	**100.0**	**0.0**

Table 5.10: Mean F1-Score values were obtained for each gesture, overall accuracy, F1 score, and FNR for each participant, considering only RF, a 2-second sliding window, and 96% overlap.

for Participant 9. This subject presents lower results for "clean" and "no gesture", but the results are still above nine-nine percent, which is still remarkably good.

Overall, the results obtained in this Subject Dependence evaluation are encouraging and led to conclusions on the effect of different variables. Although these results are not conclusive on what is needed to develop a model that may be used for a wider population with good generalization, the results may be used if that is not attainable.

5.3.3.3 Subject Independence

The performance evaluation in a subject dependent scenario is important, especially in situations where the models used in applications or systems demand the recording of new data for every new participant using the system, but that is not the intention for a system where the models under consideration in this dissertation will be used. Nevertheless, the results obtained for the subject-dependent test are relevant not only to assess the effects of the overlap, sliding window size, and classifier, but more importantly to be used if a good model generalization can not be attained.

The goal of this subsection is to explore if it is possible to end up with a good combination of classifier, window overlap, and sliding window size, to obtain a model that shows good generalization, capable of addressing multiple situations regardless of the user being considered, which is very relevant considering a real-world scenario. To achieve this, a subject independent test was conducted, which consists in testing models with data from never-seen subjects.

In this subsection, an evaluation of all classifiers will be addressed when training with all but one participant and testing with the one being left out repeating the same process for all available subjects. Similar to what was done in the subject-dependent scenario, the effect of the classifier, window overlap, and sliding window size will be evaluated to reach a good balance between all variables, allowing to obtain a good gesture recognition model, with good generalization, for the considered scenario.

5.3.3.3.1 Classifier Effect

The same five classifiers used in the subject-dependent scenario were applied in the subject-independent situation to see whether any of them stands as a superior solution for gesture recognition. Considering the results for the subject-dependent scenario, SVM and RF exhibited the strongest results, while DT exhibited the worst. Having these results in mind, and considering the way DT is built, it is expected for them to produce, once again, the worst results.

Similar to what was done in the subject-dependent scenario, the comparison of the mean accuracy and F1-score for the different classifiers is presented in Figure 5.21.

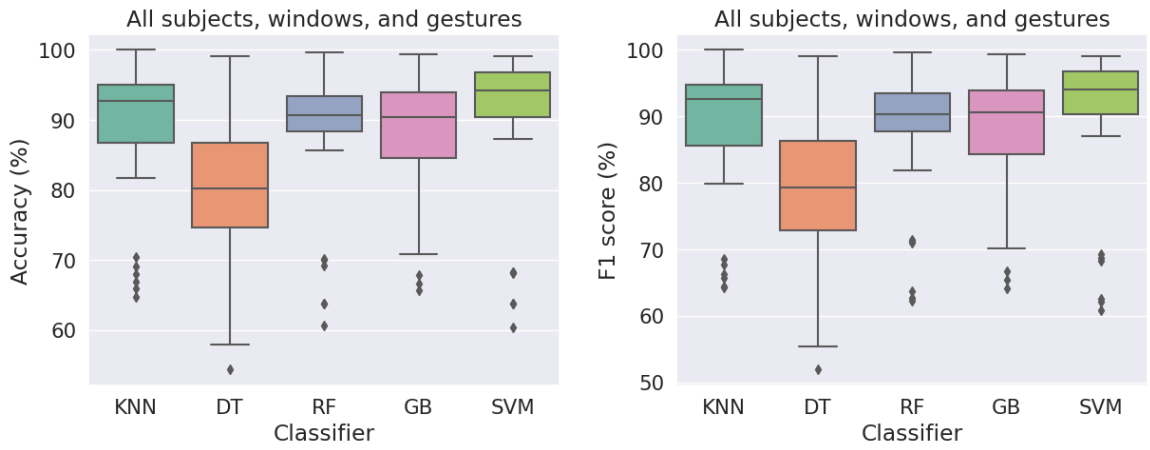


Figure 5.21: Boxplot for the Mean accuracy and F1-Score results for the classifier effect on a subject-independent scenario.

As it is observable, and as expected, DT produces the worst results by far when compared to the other four classifiers. This time, SVM produces, on average, the best results, with accuracy and F1-Score exhibiting low variability and a high median value, going in line with what was observed in the subject-dependent scenario. RF, however, although also producing low variability for accuracy, it produces a lower median value comparatively to SVM.

kNN and GB exhibit similar results, with median accuracy values above the mark of 90% but with a considerable amount of variability, however the mean accuracy median results for kNN is close to the one presented by SVM.

Since the F1-Value values are on average better and the mean accuracy values have lower variation, RF is regarded as the second-best option for classification, even though it presents a lower accuracy median value than kNN.

	DT		GB		kNN		RF		SVM	
Participant Identifier	**Mean accuracy(%)**	**Mean F1(%)**	**Mean accuracy(%)**	**Mean F1(%)**	**Mean accuracy(%)**	**Mean F1(%)**	**Mean accuracy(%)**	**Mean F1(%)**	**Mean accuracy(%)**	**Mean F1(%)**
1	87.3	86.8	93.6	93.6	92.9	92.8	96.9	96.9	96.6	96.6
2	79.9	78.9	92.6	92.7	87.7	86.5	92.1	91.5	94.1	93.8
3	58.5	56.7	69.3	68.2	67.5	66.2	66.3	67.0	65.5	65.3
4	96.8	96.8	99.1	99.1	95.9	95.9	99.3	99.3	97.9	97.9
5	74.1	73.3	84.6	84.1	94.7	94.5	89.6	89.2	94.4	94.4
6	83.8	83.5	95.8	95.8	94.2	94.1	92.0	92.0	98.8	98.8
7	86.6	86.5	90.1	90.5	90.7	90.5	92.1	92.4	91.4	91.4
8	74.8	73.7	83.2	81.6	87.4	87.0	89.9	89.4	88.9	88.6
9	80.3	79.3	86.8	85.1	82.9	81.6	89.9	89.5	92.6	92.4
10	76.4	72.7	89.5	89.2	98.5	98.5	86.5	83.7	95.9	95.8

Table 5.11: Classifier effect over each participant on a subject independence scenario.

When considering the results presented in Table 5.11, it gets more clear that SVM and RF have the best overall results. Although in one case kNN shows the best accuracy results (*e.g.,* participant 10), SVM and RF consistently present good results, being stable for the classification of the movements being considered and therefore being considered the best options for the scenario.

5.3.3.3.2 Window Size and Overlap Effect

The results obtained for the comparison of different overlaps and windows are presented in Figures 5.22, 5.23 and 5.24, which contain the mean accuracy, F1-Score, and False Negative Rate (FNR) for the "No Gesture" over each different combination of classifier, overlap window, and sliding window size.

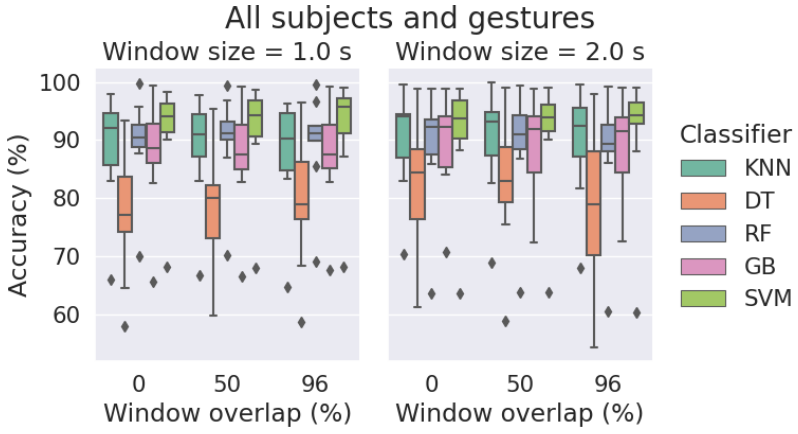


Figure 5.22: Boxplot for mean accuracy values considering the overlap and window effect over each different classifier, for the subject independent test.

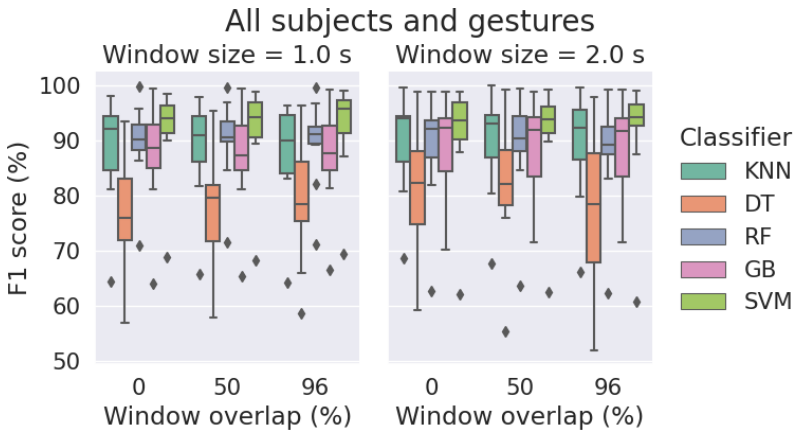


Figure 5.23: Boxplot for mean F1-Score values considering the overlap and window effect over each different classifier, for the subject independent test.

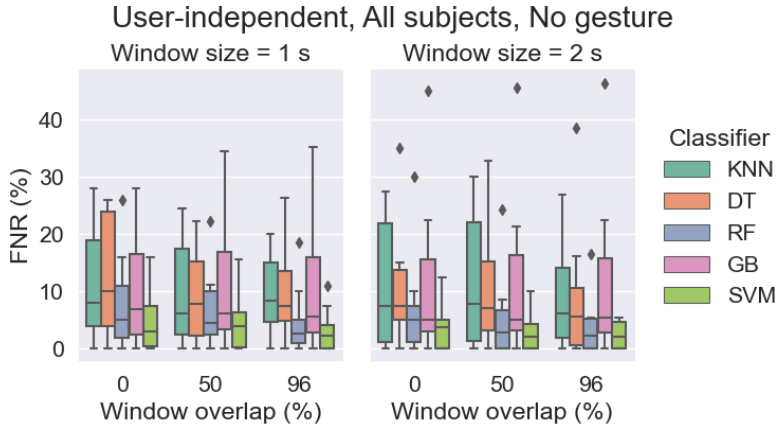


Figure 5.24: Boxplot for the False-negative rate (FNR) for "No gesture" calculated over all combinations of classifier, window overlap, and sliding window size.

When considering only the effect of different overlaps, what is observed is that different overlaps do not always present a considerable difference. RF, for instance, gets overall less variability on the Mean accuracy and F1-Score with a higher overlap, but in the case of the 2-second sliding window, the median lowers.

DT on the other hand produces better median results on a 1-second sliding window with a 50% window overlap compared to a 0% or 96%, but the latter shows better results in terms of variability. In the case of DT with a 2-second sliding window, the results are overall better with a 0% overlap.

SVM presents better results with bigger overlaps when considering all three metrics (accuracy, F1-Score, and FNR for the "No Gesture").

Regarding the sliding window size, there are a few observations to be made. RF classifier holds worse results when changing from a 1-second sliding window to a 2-second sliding window, mainly due to more variability and lower medians, while SVM presents the opposite behavior, getting overall less variability when going from a 1 to the 2-second sliding window.

Figure 5.25 presents the results for accuracy and F1-Score for different window sizes and

overlaps when considering only SVM, which was the classifier that yielded the best results when considering only the effect of the classifier.

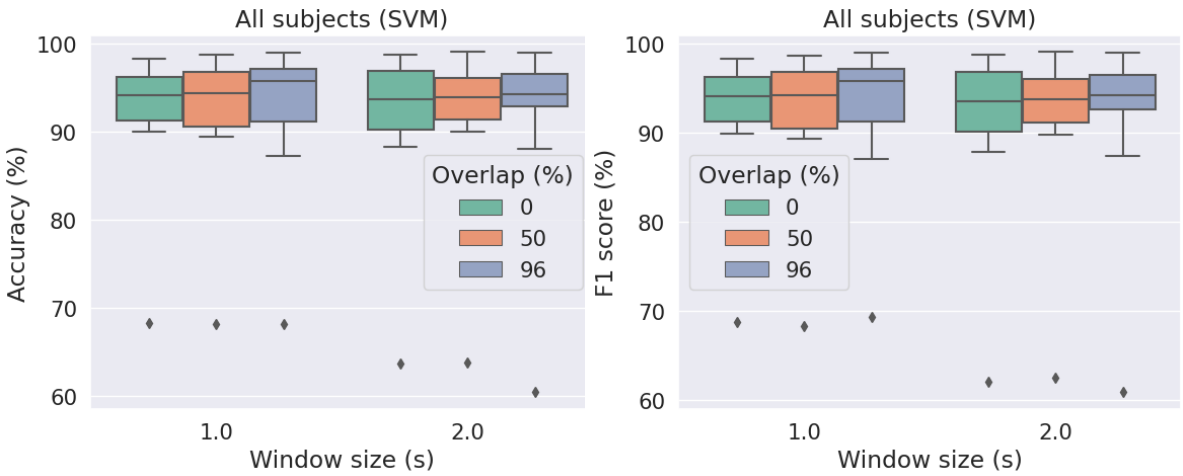


Figure 5.25: Mean Accuracy and F1-Score values for different combinations of sliding window size and overlap considering only SVM, for the subject independent scenario.

When taking into consideration only the results from the SVM, regarding the sliding window, there is not an option that clearly outperforms the others when considering the accuracy and F1 score. However, the False Negative Rate results for the "No gesture" are better with a 2-second sliding window size and 96% window overlap (see Fig. 5.24), therefore that is preferable.

In essence, there is not a relation between the overlap selected and classification correctness. Similar to the subject-dependent scenario, both sliding window sizes are viable.

5.3.3.3.3 Results per Gesture

With the effects of classifiers, overlaps, and sliding windows discussed, it was determined that the best combination results from the use of SVM, and a 2-second sliding window with a 96% overlap.

Similarly to what was analyzed in the subject dependent scenario, to understand how the combination of variables work on the recognition of each gesture, the F1-Score results for each participant using the combination of SVM, 2-second window size, and a 96% overlap were calculated and are presented in Table 5.12.

Participant Identifier	Knock F1(%)	Twist F1(%)	Clean F1(%)	Circle F1(%)	Come F1(%)	No gesture F1(%)	Accuracy(%)	F1-Score(%)	FNR (no gesture)(%)
1	94.7	93.4	98.5	100.0	99.6	90.8	96.1	96.2	3.2
2	100.0	74.9	88.4	93.0	100.0	99.5	93.1	92.6	1.0
3	97.9	41.4	79.8	60.9	37.3	47.9	60.4	60.9	0.0
4	100.0	93.9	99.9	95.3	100.0	97.9	97.8	97.8	4.2
5	98.1	89.8	97.6	94.7	100.0	86.6	94.4	94.5	4.9
6	99.7	97.5	100.0	100.0	99.5	97.2	98.9	98.9	5.4
7	94.1	99.6	97.2	85.6	99.6	88.2	94.1	93.9	0.0
8	100.0	94.2	63.2	95.7	89.9	81.7	88.1	87.4	0.0
9	97.4	85.9	77.9	97.6	99.8	97.3	92.8	92.7	5.3
10	100.0	88.9	100.0	100.0	99.7	91.2	96.7	96.7	0.0
Total	**98.2**	**85.9**	**90.3**	**92.3**	**92.5**	**87.8**	**91.3**	**91.2**	**2.4**

Table 5.12: Mean F1-Score values obtained for each gesture, overall accuracy, F1 score, and FNR for each participant, considering only SVM, a 2-second sliding window, and 96% overlap.

The results presented in Table 5.12 show some important details. When comparing the different subjects, we can observe that only one has an overall accuracy and F1-Score below 87% percent, and the remaining subjects present relatively high metric values. The "Knock" gesture exhibited the best results considering all users. The remaining gestures exhibited overall F1-Scores above 90% except for the "Twist" and "No Gesture", but that is mostly due to participant 2 that held the worst results.

The Confusion Matrix presented in Figure 5.26 provides for a more precise understanding of which gestures are being misunderstood as others and, as a result, a more detailed understanding of which gestures are better recognized. The values displayed in the confusion matrix presented in Figure 5.26 correspond to the sum of the number instances considering all subjects (the diagonal values were set to 0).

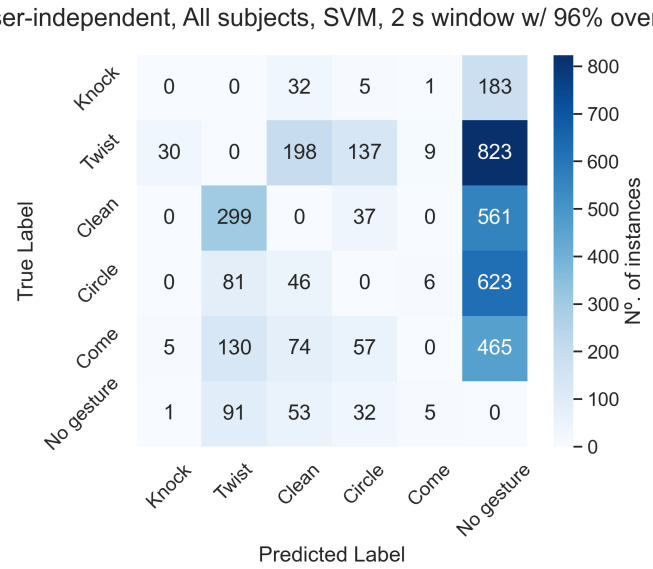


Figure 5.26: Confusion matrix for the subject-independent case, when using SVM and 2-second window overlap with 96% overlap, considering all subjects.

In terms of gestures, we can observe from the confusion matrix that the majority of gestures are misinterpreted as "No gesture", with the "Twist" gesture being the worst case,

presenting the most cases where the gesture is confused as a "No gesture". These results are most likely due to the number of movements included in the "No gesture" class, which encompasses not only the lack of movement but also tiny motions that are commonly performed in a bed setting (e.g. bringing the hand up to the face). On the other side, the "No gesture" is not as commonly confused with other gestures, with the "Twist" gesture having the largest frequency of false negatives, when considering the "No gesture" as the positive class. Overall, the gesture with the best results is "Knock", followed by "Circle" and "Come", which is confirmed by the F1-Score values presented in Table 5.12.

To conclude, the combination of SVM, 2-second window size, and a 96% overlap, holds, for the subject independent scenario, very good classification results, with at least three of the gestures being detected with very good results overall, and therefore showing promising results for the attainment of a generalized model capable of providing great results for the recognition of the predefined gestures considered for the system.

5.4 Conclusion

The two experimental procedures conducted in this chapter help understand how machine learning techniques behave for gesture recognition in a bed environment with the user laying in bed.

Although the preliminary experiment was conducted with a not-so-optimal setup and number of participants, the results were encouraging for the development of the second setup and a few observations led to believe that the execution of gestures in a bed scenario was viable. The second experiment, with an enhanced setup and a more improved data collection process allowed to assess, with more certainty, the performance of machine learning techniques for the recognition of gestures. The results are good even when considering a subject-independent scenario, which is important for the goal of this dissertation.

Although not optimal, due to the limitations in which the involvement of users was constrained by the COVID-19 pandemic, no aphasics could be recruited to retrieve gesture data. Nevertheless, the sample from both experiments shows great diversity, which is important to enrich the dataset with different executions of the proposed movements, leading to a more realistic scenario with the possibility of obtaining a better and more prepared model for classification on a wider population beyond the one selected for the experiments. From the process of data gathering, the most important result that was taken is that all participants were comfortable executing the gestures, which is very important considering that in some cases these gestures might be performed by people with some condition that affects their performance of the gestures, therefore, having gestures simple to understand and execute is important.

Considering the classification results for the second experiment, only, when assessing the user-dependent solution, where a model is trained from data from a given subject and used to recognize gestures performed by that same subject, the algorithms that led to be best results were RF and SVM, with almost perfect identification of the gestures. The results were already expected due to it being trained and tested for each user individually, but the results are nonetheless important if a model generalization is not attainable and therefore a model for each user using the system needs to be trained.

For the user-independent solution, the results were also very encouraging with SVM leading to the best results overall, with most participants presenting an overall accuracy and F1-Score over 92%, with at least three of the gestures being recognized almost flawlessly,

which was stated as being important when considering the gesture selection in Section 5.1.

When comparing the performance of user-dependent and independent solutions, the latter performs poorly. This outcome was already expected, as it is more difficult to classify the movements of a never-seen subject using a model trained on data from other subjects, due to normal differences in how different users execute the gestures. However, while a user-dependent approach allows for very high gesture recognition performance, it has the drawback of requiring data collection from each new user using the system, which is neither convenient nor cost-effective. However, the results indicate that a user-dependent is not needed and that a good model generalization may be obtained with more data for each gesture and more participants performing them.

Chapter 6

Conclusions

6.1 Work Summary

The work developed in this dissertation followed a specific path to fulfill the proposed goals. Therefore, an investigation on new ways for people with aphasia to communicate in a bedroom scenario while laying in bed was conducted and the processes followed for the execution of it were as follows:

1. Acquisition of knowledge in the problem domain, concerning aphasia and existing solutions, to support this audience. This enabled us to define target scenarios and formulate the requirements for a communication support solution in the bed scenario.

2. Hands-on exploration of different configurations concerning the positioning of sensors to find a solution that allowed to tackle the project goal.

3. Definition, based on related work in the area of gesture recognition, of a set of dynamic arm movements that could be used as a non-verbal form of communication and that would take into account the environment in which they will be used.

4. Experimentation of widely known machine learning techniques together with data captured from sensors placed on participants' wrists to recognize a small set of gestures, to support communication and provide a new way to communicate. This early work gave confidence to later data gathering from a wider group, to find the best model possible for the goal of this dissertation.

5. Definition of the final architecture of the system and implementation of a prototype that captures and distinguishes a predefined set of gestures and sends requests to caregivers' smartphones using a minimal and non-obtrusive setup.

6. Improvement of the setup with alternative interactions, which include the possibility of the caregiver to send yes/no questions to the aphasic to which he/she may respond with two of the gestures supported.

The steps followed allowed for the completion of all of the dissertation's objectives, which was critical to produce a coherent work with a few key contributions for the scenario under consideration.

6.2 Main Results and Contributions

The work developed in this dissertation has two key contributions: the investigation done on gesture recognition in a bed scenario using machine learning techniques which yielded great and promising results, and the proof-of-concept system developed that establishes a bidirectional communication channel between a user and a caregiver responsible for that person, supported by the use of gestures by the user.

The gesture recognition investigation that was conducted in this dissertation is important for the fact that it provided contributions for gesture recognition in an in-bed scenario which was lacking in the topic of gesture recognition. The results gave a great output of what could be expected when using a small but considerable set of gestures, which were chosen taking into consideration the bed environment. A subject-dependent scenario was first conducted and offered good results. In the real world, a solution relying that each user executes gestures for training purposes is not practical. However, the results obtained when considering a user-independent scenario were promising and showed that a model generalization may be attained, making both approaches viable with the second being preferred.

The system developed contemplated the feature of gesture recognition, that was built around the investigation done for this dissertation. Gestures are used to trigger alerts that are sent to the caregiver through its smartphone and allow bidirectional communication between both parts. This is an important contribution to the project but further improvements need to be done, nevertheless, it was built considering that further modules could be introduced. For that, messaging methods that allow the introduction of new modules without the need of restructuring the whole system was chosen. Preliminary tests of the system presented good performance with little to no delay on the responses to gestures executed, which is important considering a use case of emergency.

Although the system was built around the context of people with aphasia, the prototype and solution developed are not limited to those. The system is simple enough to be understood by the majority of people and can be used by anyone that needs a solution to connect to a caregiver for basic assistance, without the need to call or read from a screen.

6.3 Future Work

While the work developed in this dissertation presented good results, the system and performed research are just preliminary results that serve as a baseline for what may be done regarding the scenario and population being addressed.

The prototype developed is a first attempt at a system that uses gestures to be used for communication. Although an initial experiment was carried out, the approach was brief and did not include an in-depth evaluation of the system's usability by the target population, performance, or energy efficiency. As a result, future research should focus on evaluating the system's with the target group, particularly with older people who are more likely to be affected by these speech disabilities, to determine whether the system is viable and if it can be extended to be used in other scenarios, such as interacting with smart home systems, and even be used outside of the bedroom environment, thereby providing more support to this population throughout the day.

Future research should also contemplate the development and improvement of features of the system as well as the extension of the research work done regarding gesture recognition.

The most important aspects to be addressed are the following:

1. Extensive assessment of the system with final users in regards to usability, performance, and energy efficiency;

2. Gathering of more gesture data, with the participation of a larger group. This evaluation could be extended by gathering data from a single user on different days, so the performance of classification could be evaluated taking into consideration the variations that may occur from one day to another on the same participant.

3. Implement new interaction possibilities on the caregiver side, with more responses and question alternatives. This step should take feedback from final users and their caregivers to understand their needs and therefore offer a more concise solution.

4. Extension of the system regarding customization options, with, for instance, the possibility of choosing and changing what each gesture means for the user, and provide other visualization features that allow a better follow-up from the caregiver.

5. Extend the system with the integration of in-bed sensors that can provide alternative interaction possibilities or enhancement of those already implemented;

6. Extend the speaker service by implementing more feedback interactions, e.g., when an aphasic successfully executes a gesture a confirmation of the success could be broadcast on the speaker. This could not only further minimize false positives (reaching the smartphone), but also, and mainly, reinforce the comfort in the communication of the person with aphasia.

This foreseen future work offers new lines for development, but we also expect that new requirements would come from working with final users and professionals in the field, therefore, getting feedback from both parties is crucial in determining which topic should be addressed first.

References

[1] AR Damasio. Aphasia. *The New England journal of medicine*, 326(8):531—539, February 1992.

[2] Aphasia definitions. `https://www.aphasia.org/aphasia-definitions/`. [Online; accessed 03-April-2021].

[3] Afonso Guimarães, Ana Patrícia Rocha, Luís Santana, Ilídio C. Oliveira, José Maria Fernandes, Samuel Silva, and António Teixeira. Enhanced communication support for aphasia using gesture recognition: The bedroom scenario. In *2021 IEEE International Smart Cities Conference (ISC2)*, pages 1–4, 2021.

[4] Aphasia statistics. `https://www.aphasia.org/aphasia-resources/aphasia-statistics/7`. [Online; accessed 05-May-2021].

[5] Ardila Benson. In *Aphasia: A clinical perspective*. Oxford University Press, 1996.

[6] Margaret Rogers, J. King, and Nancy Alarcon. *Proactive management of primary progressive aphasia*, pages 305–337. 01 2000.

[7] J. Lyon. Coping with aphasia. 1998.

[8] Julius Fridriksson, Dirk-Bart den Ouden, Argye E Hillis, Gregory Hickok, Chris Rorden, Alexandra Basilakos, Grigori Yourganov, and Leonardo Bonilha. Anatomy of aphasia revisited. *Brain*, 141(3):848–862, 01 2018.

[9] Marcelo L. Berthier. Poststroke aphasia. *Drugs & Aging*, 22(2):163–182, Feb 2005.

[10] Yasmin Elsahar, Sijung Hu, Kaddour Bouazza-Marouf, David Kerr, and Annysa Mansor. Augmentative and alternative communication (AAC) advances: A review of configurations for individuals with a speech disability. *Sensors*, 19(8), 2019.

[11] Karyn Moffatt, Golnoosh Pourshahid, and Ronald M. Baecker. Augmentative and alternative communication devices for aphasia: The emerging role of "smart" mobile devices. *Univers. Access Inf. Soc.*, 16(1):115–128, March 2017.

[12] Sarah E. Wallace, Karen Hux, Kelly Knollman-Porter, Jessica A. Brown, Elizabeth Parisi, and Rebecca Cain. Reading behaviors and text-to-speech technology perceptions of people with aphasia. *Assistive Technology*, 0(0):1–12, 2021. PMID: 33724912.

[13] Tobii Dynavox. Eye tracking/Eye gaze access and speech generating devices (SGD). `https://www.tobiidynavox.com/products/devices`. [Online; accessed 03-April-2021].

[14] Talk to Me Technologies. Eyespeak communication systems. https://www.tobiidynavox.com/products/devices. [Online; accessed 06-June-2021].

[15] Alea Technologies. IntelliGaze. https://www.intelligaze.com/en/. [Online; accessed 03-April-2021].

[16] Lingraphica. Lingraphica. https://www.aphasia.com/. [Online; accessed 03-April-2021].

[17] Intuary. Verbally for iPad. http://verballyapp.com/about_us.html. [Online; accessed 06-June-2021].

[18] AssistiveWare. Proloquo2Go - AAC app with symbols. https://www.assistiveware.com/products/proloquo2g. [Online; accessed 10-May-2021].

[19] Lingraphica. Smalltalk. https://www.aphasia.com/smalltalk-aphasia-apps/. [Online; accessed 04-April-2021].

[20] Therapy Box. Predictable. https://therapy-box.co.uk/predictable. [Online; accessed 03-April-2021].

[21] Hina Shaheen and Tariq Mehmood. Talking gloves: Low-cost gesture recognition system for sign language translation. In *2018 IEEE Region Ten Symposium (Tensymp)*, pages 219–224, 2018.

[22] Jesús Galván-Ruiz, Carlos M. Travieso-González, Acaymo Tejera-Fettmilch, Alejandro Pinan-Roescher, Luis Esteban-Hernández, and Luis Domínguez-Quintana. Perspective and evolution of gesture recognition for sign language: A review. *Sensors*, 20(12), 2020.

[23] Tijana Vuletic, Alex Duffy, Laura Hay, Chris McTeague, Gerard Campbell, and Madeleine Grealy. Systematic literature review of hand gestures used in human computer interaction interfaces. *International Journal of Human-Computer Studies*, 129:74–94, 2019.

[24] Munir Oudah, Ali Al-Naji, and Javaan Chahl. Hand gesture recognition based on computer vision: A review of techniques. *Journal of Imaging*, 6(8), 2020.

[25] Shahzad Ahmed, Karam Dad Kallu, Sarfaraz Ahmed, and Sung Ho Cho. Hand gestures recognition using radar sensors for human-computer-interaction: A review. *Remote Sensing*, 13(3), 2021.

[26] Hongyi Liu and Lihui Wang. *Latest Developments of Gesture Recognition for Human–Robot Collaboration*, pages 43–68. Springer International Publishing, Cham, 2021.

[27] Chutisant Kerdvibulvech. A review of augmented reality-based human-computer interaction applications of gesture-based interaction. In Constantine Stephanidis, editor, *HCI International 2019 – Late Breaking Papers*, pages 233–242, Cham, 2019. Springer International Publishing.

[28] Hira Ansar, Ahmad Jalal, Munkhjargal Gochoo, and Kibum Kim. Hand gesture recognition based on auto-landmark localization and reweighted genetic algorithm for healthcare muscle activities. *Sustainability*, 13(5), 2021.

[29] Aurelijus Vaitkevičius, Mantas Taroza, Tomas Blažauskas, Robertas Damaševičius, Rytis Maskeliūnas, and Marcin Woźniak. Recognition of american sign language gestures in a virtual reality using leap motion. *Applied Sciences*, 9(3), 2019.

[30] Kathrin Kefer, Clemens Holzmann, and Rainhard Dieter Findling. Evaluating the placement of arm-worn devices for recognizing variations of dynamic hand gestures. *J. Mob. Multimed.*, 12(3–4):225–242, April 2017.

[31] Marc Kurz, Robert Gstoettner, and Erik Sonnleitner. Smart rings vs. smartwatches: Utilizing motion sensors for gesture recognition. *Applied Sciences*, 11(5), 2021.

[32] Yu Ling, Xiang Chen, Yuwen Ruan, Xu Zhang, and Xun Chen. Comparative study of gesture recognition based on accelerometer and photoplethysmography sensor for gesture interactions in wearable devices. *IEEE Sensors Journal*, 21(15):17107–17117, 2021.

[33] Hongyang Zhao, Shuangquan Wang, Gang Zhou, and Daqing Zhang. Ultigesture: A wristband-based platform for continuous gesture control in healthcare. *Smart Health*, 11:45–65, 2019.

[34] Trung-Hieu Le, Thanh-Hai Tran, and Cuong Pham. The internet-of-things based hand gestures using wearable sensors for human machine interaction. In *2019 International Conference on Multimedia Analysis and Pattern Recognition (MAPR)*, pages 1–6, 2019.

[35] Meng-Kun Liu, Yu-Ting Lin, Zhao-Wei Qiu, Chao-Kuang Kuo, and Chi-Kang Wu. Hand gesture recognition by a mmg-based wearable device. *IEEE Sensors Journal*, 20(24):14703–14712, 2020.

[36] Khanh Nguyen-Trong, Hoai Nam Vu, Ngon Nguyen Trung, and Cuong Pham. Gesture recognition using wearable sensors with bi-long short-term memory convolutional neural networks. *IEEE Sensors Journal*, 21(13):15065–15079, 2021.

[37] Peide Zhu, Hao Zhou, Shumin Cao, Panlong Yang, and Shuangshuang Xue. Control with gestures: A hand gesture recognition system using off-the-shelf smartwatch. In *2018 4th International Conference on Big Data Computing and Communications (BIGCOM)*, pages 72–77, 2018.

[38] Hobeom Han and Sang Won Yoon. Gyroscope-based continuous human hand gesture recognition for multi-modal wearable input device for human machine interaction. *Sensors*, 19(11), 2019.

[39] J. Guillermo Colli-Alfaro, Anas Ibrahim, and Ana Luisa Trejos. Design of user-independent hand gesture recognition using multilayer perceptron networks and sensor fusion techniques. In *2019 IEEE 16th International Conference on Rehabilitation Robotics (ICORR)*, pages 1103–1108, 2019.

[40] Mateus M. Luna, Thyago P. Carvalho, Fabrizzio Alphonsus A. M. N. Soares, Hugo A. D. Nascimento, and Ronaldo M. Costa. Wrist player: A smartwatch gesture controller for smart tvs. In *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*, volume 2, pages 336–341, 2017.

[41] Ananda Sankar Kundu, Oishee Mazumder, Prasanna Kumar Lenka, and Subhasis Bhaumik. Hand gesture recognition based omnidirectional wheelchair control using imu and emg sensors. *Journal of Intelligent & Robotic Systems*, 91(3):529–541, 2018.

[42] Alessandro Carfi, Carola Motolese, Barbara Bruno, and Fulvio Mastrogiovanni. Online human gesture recognition using recurrent neural networks and wearable sensors. In *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 188–195, 2018.

[43] Boris Gromov, Gabriele Abbate, Luca M. Gambardella, and Alessandro Giusti. Proximity human-robot interaction using pointing gestures and a wrist-mounted imu. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8084–8091, 2019.

[44] Lorenzo Porzi, Stefano Messelodi, Carla Mara Modena, and Elisa Ricci. A smart watch-based gesture recognition system for assisting people with visual impairments. IMMPD '13, page 19–24, New York, NY, USA, 2013. Association for Computing Machinery.

[45] Tea Marasović and Vladan Papić. Accelerometer-based gesture classification using principal component analysis. In *SoftCOM 2011, 19th International Conference on Software, Telecommunications and Computer Networks*, pages 1–5, 2011.

[46] Franck Tchuente, Natalie Baddour, and Edward D. Lemaire. Classification of aggressive movements using smartwatches. *Sensors*, 20(21), 2020.

[47] Yen-Cheng Chu, Yun-Jie Jhang, Tsung-Ming Tai, and Wen-Jyi Hwang. Recognition of hand gesture sequences by accelerometers and gyroscopes. *Applied Sciences*, 10(18), 2020.

[48] Oresti Banos, Juan-Manuel Galvez, Miguel Damas, Hector Pomares, and Ignacio Rojas. Window size impact in human activity recognition. *Sensors*, 14(4):6474–6499, 2014.

[49] Belkacem Chikhaoui, Bing Ye, and Alex Mihailidis. Ensemble learning-based algorithms for aggressive and agitated behavior recognition. pages 9–20, 11 2016.

[50] Akbar Dehghani, O. Sarbishei, Tristan Glatard, and Emad Shihab. A quantitative comparison of overlapping and non-overlapping sliding windows for human activity recognition using inertial sensors. *Sensors*, 19:5026, 11 2019.

[51] Peide Zhu, Hao Zhou, Shumin Cao, Panlong Yang, and Shuangshuang Xue. Control with gestures: A hand gesture recognition system using off-the-shelf smartwatch. In *2018 4th International Conference on Big Data Computing and Communications (BIGCOM)*, pages 72–77, 2018.

[52] Nabeel Siddiqui and Rosa H. M. Chan. Multimodal hand gesture recognition using single imu and acoustic measurements at wrist. *PLOS ONE*, 15(1):1–12, 01 2020.

[53] Uzma Abid Siddiqui, Farman Ullah, Asif Iqbal, Ajmal Khan, Rehmat Ullah, Sheroz Paracha, Hassan Shahzad, and Kyung-Sup Kwak. Wearable-sensors-based platform for gesture recognition of autism spectrum disorder children using machine learning algorithms. *Sensors*, 21(10), 2021.

[54] Yong-Ting Wang and Hsi-Pin Ma. Real-time continuous gesture recognition with wireless wearable imu sensors. In *2018 IEEE 20th International Conference on e-Health Networking, Applications and Services (Healthcom)*, pages 1–6, 2018.

[55] Edwin Valarezo Añazco, Seung Ju Han, Kangil Kim, Patricio Rivera Lopez, Tae-Seong Kim, and Sangmin Lee. Hand gesture recognition using single patchable six-axis inertial measurement unit via recurrent neural networks. *Sensors*, 21(4), 2021.

[56] Kalpana Lamb and Swati Madhe. Automatic bed position control based on hand gesture recognition for disabled patients. pages 148–153, 09 2016.

[57] Eldad Holdengreber, Roi Yozevitch, and Vitali Khavkin. Intuitive cognition-based method for generating speech using hand gestures. *Sensors*, 21(16), 2021.

[58] Rúbia E. O. Schultz Ascari, Roberto Pereira, and Luciano Silva. Computer vision-based methodology to improve interaction for people with motor and speech impairment. *ACM Trans. Access. Comput.*, 13(4), October 2020.

[59] Rikke Friis Dam and Teo Yu Siang. Personas – a simple introduction. `https://www.interaction-design.org/literature/article/personas-why-and-how-you-should-use-them`.

[60] Moein Enayati, Marjorie Skubic, James M. Keller, Mihail Popescu, and Nasibeh Zanjirani Farahani. Sleep posture classification using bed sensor data and neural networks. In *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 461–465, 2018.

[61] Bo Yu Su, Moein Enayati, K. C. Ho, Marjorie Skubic, Laurel Despins, James Keller, Mihail Popescu, Giovanna Guidoboni, and Marilyn Rantz. Monitoring the relative blood pressure using a hydraulic bed sensor system. *IEEE Transactions on Biomedical Engineering*, 66(3):740–748, 2019.

[62] Qingju Liu, Mark Kenny, Ramin Nilforooshan, and Payam Barnaghi. An intelligent bed sensor system for non-contact respiratory rate monitoring, 2021.

[63] Miika Kalske, Niko Mäkitalo, and Tommi Mikkonen. *Challenges When Moving from Monolith to Microservice Architecture*, pages 32–47. 02 2018.

[64] Wear OS. `https://wearos.google.com/`, 2021. [Online; accessed 02-September-2021].

[65] Raspberry Pi Foundation. Raspberry Pi. `https://www.raspberrypi.org/`, 2021. [Online; accessed 14-September-2021].

[66] Scheduled Executor Service, (Java Platform SE 7). `https://docs.oracle.com/javase/7/docs/api/java/util/concurrent/ScheduledExecutorService.html`. [Online; accessed 10-June-2021].

[67] Ryan Govostes. Pybluez. `https://github.com/pybluez/pybluez`. [Online; accessed 10-May-2021].

[68] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[69] Padraig Cunningham, Bahavathy Kathirgamanathan, and Sarah Jane Delany. Feature Selection Tutorial with Python Examples, 06 2021.

Appendix A

Recording Application

To allow the recording and retrieval of data for training and testing purposes on the second experiment conducted, two applications were developed, one for the Wear OS Smartwatch and the other for an Android Smartphone.

The first application, implemented directly on the smartwatch, is responsible for collecting and sending sensor data. To control the recordings, this Wear OS application receives commands from an Android Smartphone device, which has an application for that purpose.

To provide communication between the two devices, messages are exchanged. With Wear OS by Google, a wearable has multiple ways to send and sync data. The "Wearable Data Layer API", which is part of Google Play services, was used for this project (take into account that for this to work, both devices must be linked). With the Wearable Data Layer API different clients can be used:

1. **CapabilityClient** offers details on which Wear OS network nodes support certain custom app features. Nodes can be either wearables or smartphones and capabilities are device defined;

2. **MessageClient** useful for remote procedure calls like managing a handheld media player from the wearable.

3. **ChannelClient** used for the transfer of larger files or data that can not be shared using **MessageClient** and data streams such as voice.

4. **DataClient** offers an API for components to read and write to a DataItem that is shared across all Wear OS devices.

For this recording application, **DataClient** was chosen. As stated, DataClient uses DataItems to synchronize information between handhelds and wearables and comprises of a payload (the data being sent between devices) and a path (a unique path where the data is written and where the data can be read by other devices listening to that path). Consecutive DataClient messages must have distinct payloads, therefore messages must be provided with something that makes them unique. As a result, a timestamp is appended to all payloads, making them unique.

To start a recording, the user taps the "Record" button and the handheld sends a start message together with a time limit indicating how long the recording will last, this value ranges from 1 to 10 seconds and can be selected using a SeekBar on the mobile device.

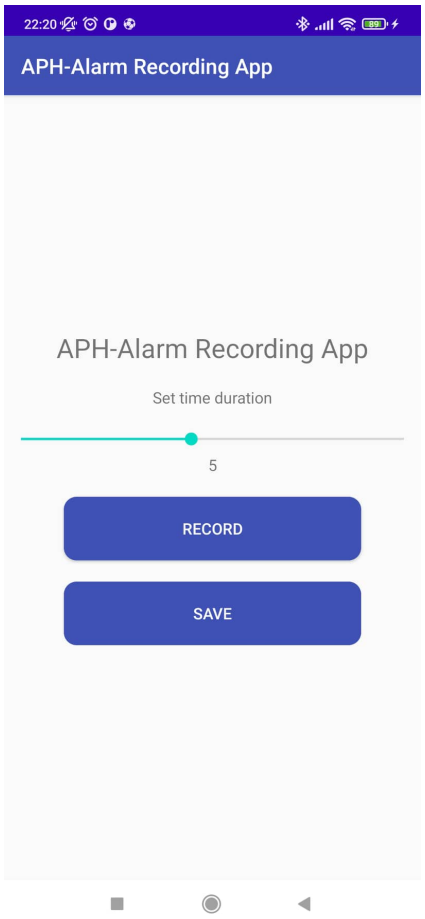


Figure A.1: Starting screen.

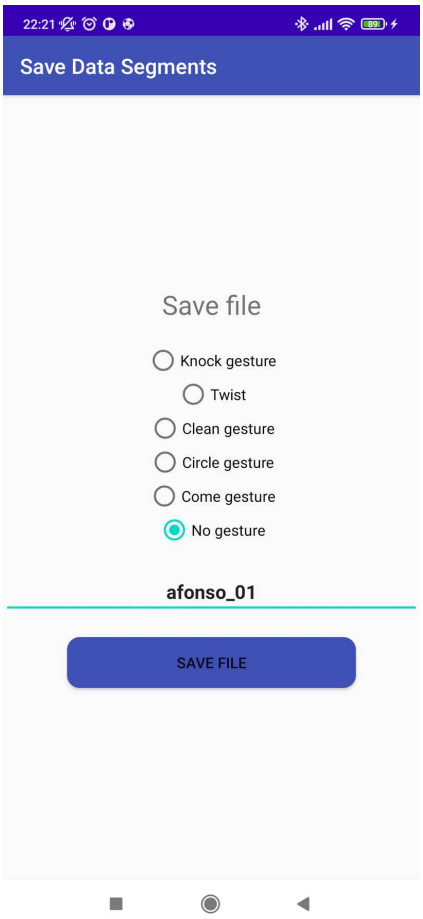


Figure A.2: Save data screen.

Although a set amount of time is defined at the start, a recording can, at any moment be stopped by sending a message indicating it. When this occurs the handheld receives the data that was recorded until that moment, otherwise, it will record for the time indicated, and only then does the information gets sent.

When the smartwatch application receives a start command, it begins filling a Comma-separated values(CSV) object. This object is composed of data records where each one contains for a given time the values of the sensor data from the accelerometer, gyroscope, magnetometer, along with a timestamp. When the time given for a recording is completed or a stop command is received, the structure is sent back to the mobile smartphone that started the recording utilizing a DataItem through the DataClient API.

The data received may then be saved or not on the mobile device's internal storage. Each file will be saved in a folder appropriate to each participant and each gesture, for example, **"/data/001/twist/"** holds the files for the twist gestures by the participant with identifier '001'.

The sampling frequency of the Android Wear smartwatch was set to fifty Hertz in the application[1], however owing to Android's internal sensor implementation where the polling

[1]Using a Java Executor with a scheduler that executes approximately every twenty milliseconds and saves

rate cannot be set, it may vary a small amount.

the sensor information