ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA DE SISTEMAS

REAL-TIME HAND GESTURE RECOGNITION USING MACHINE LEARNING AND INFRARED INFORMATION

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE DOCTOR EN INFORMÁTICA

RUBEN EDUARDO NOGALES PORTERO

ruben.nogales@epn.edu.ec

DIRECTOR: MARCO ENRIQUE BENALCAZAR PALACIOS

marco.benalcazar@epn.edu.ec

Quito, June 2024

ESCUELA POLITÉCNICA NACIONAL

TESIS

Para la obtención del título de

DOCTOR EN INFORMÁTICA

Resolución RPC-SO-43-No.501-2014

del Consejo de Educación Superior

Presentada por

RUBEN EDUARDO NOGALES PORTERO

Tesis dirigida por

MARCO ENRIQUE BENALCAZAR PALACIOS,

Profesor de la Escuela Politécnica Nacional (Ecuador)

REAL-TIME HAND GESTURE RECOGNITION USING MACHINE LEARNING AND INFRARED INFORMATION

Examen oral en **FECHA,**

por la siguiente comisión:

Miguel Alfonso Flores Sánchez, Ph.D.

Escuela Politécnica Nacional, President

Gustavo Javier Meschino, Ph.D.

Universidad Nacional de Mar del Plata, Secretary, external examiner

Colón Enrique Pelaez Jarrin, Ph.D.

Escuela Superior Politécnica del Litoral, External examiner

José Francisco Lucio Naranjo, Ph.D.

Escuela Politécnica Nacional, Internal member

Carlos Alberto Almeida Rodríguez, Ph.D.

Escuela Politécnica Nacional, Internal member

DECLARACIÓN

Yo, RUBEN EDUARDO NOGALES PORTERO, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

Firmado electrónicamente por:
RUBEN EDUARDO
NOGALES PORTERO

(RUBEN EDUARDO NOGALES PORTERO)

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por RUBEN EDUARDO NOGALES PORTERO, bajo mi supervisión.

(Ph.D. MARCO ENRIQUE BENALCAZAR PALACIOS)

ADVISOR

DEDICATORIA

Dedico esta tesis, con profundo amor y agradecimiento, a mi esposa Ibeth Del Salto. Su paciencia infinita, su comprensión y su apoyo incondicional han sido pilares fundamentales durante este largo y desafiante camino. Ibeth, has sido mi refugio y mi fortaleza, soportando las largas noches de estudio, las ausencias en fines de semana y feriados, y nunca dejando de creer en mí. Esta tesis es tanto tuya como mía, y sin tu presencia a mi lado, este logro no habría sido posible.

A mis queridos hijos, Luis e Ibeth Nogales Del Salto, quienes han mostrado una inmensa comprensión y madurez al aceptar mis ausencias y mi dedicación al estudio. Luis e Ibeth, su amor y su paciencia me han dado la fuerza necesaria para seguir adelante en los momentos más difíciles. Ustedes son mi mayor motivación y la razón por la cual nunca me rendí. Espero que este logro les sirva de inspiración para perseguir sus propios sueños con la misma pasión y determinación.

A mis padres, cuyo constante apoyo y amor incondicional me han acompañado a lo largo de este viaje. Han estado a mi lado en cada paso, brindándome el aliento y la fortaleza necesarios para superar los obstáculos. Sus sacrificios y enseñanzas han sido la base sobre la cual he construido este logro. Gracias por creer en mí y por estar siempre presentes.

A mi hermana Soledad y a toda su familia, que con su ejemplo de responsabilidad y superación han sido una fuente constante de inspiración. Soledad, tu dedicación, esfuerzo y fortaleza me han mostrado lo que se puede lograr con determinación y perseverancia. A mi amigo Hernán Naranjo, por su lealtad y apoyo inquebrantable, siempre dispuesto a ofrecer una palabra de aliento o una mano amiga en los momentos de necesidad. A todos ustedes, les dedico esta tesis con todo mi cariño y gratitud, reconociendo el papel esencial que han jugado en la culminación de este sueño.

AGRADECIMIENTOS

Agradezco profundamente a la Escuela Politécnica Nacional por brindarme la oportunidad de cursar el programa de doctorado en Informática. Su prestigio y excelencia académica han sido fundamentales en mi formación y desarrollo profesional. De igual manera, expreso mi más sincero agradecimiento al Dr. Marco Benalcázar, a todos los integrantes del Laboratorio de Investigación en Inteligencia y Visión Artificial “Alan Turing”, cuyo invaluable apoyo, orientación y mentoría han sido esenciales para alcanzar este importante logro. Su dedicación y paciencia durante todo el proceso de mi doctorado han sido una inspiración constante.

Asimismo, quiero expresar mi gratitud a la Universidad Técnica de Ambato por proporcionarme las facilidades necesarias para cursar mis estudios, permitiéndome conjugar mis responsabilidades profesionales con mi formación académica. Agradezco también a todos mis amigos y compañeros de oficina, quienes con su aliento y compañía diaria me han motivado y empujado a superar cada obstáculo. Su apoyo incondicional ha sido un pilar fundamental en la culminación de este gratificante proceso.

Table of Contents

List of Figures

Table Index

ABSTRACT

Working on hand gesture recognition is important as it enables seamless human-computer interaction, enhances accessibility in various domains, and opens avenues for innovative applications in fields such as healthcare, gaming, and virtual reality. This thesis explores the field of hand gesture recognition, utilizing infrared information and a set of machine learning and deep learning algorithms. Within the domain of machine learning, a diverse set of algorithms are evaluated, including Artificial Neural Network (ANN), Support Vector Machine (SVM), k-Nearest Neighbors (KNN), and Decision Trees (DT). We also use the following deep learning algorithms: Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and bidirectional Long-Short-Term Memory (BiLSTM), aiming to determine their potential to improve hand gesture recognition performance. The thesis commences by addressing the intricacies of hand gesture recognition. This task is far from trivial, as it involves complex challenges associated with classification and pattern recognition. Among these challenges are the variability in hand shapes and sizes, occlusions, dynamic movement patterns, and the necessity for robustness to diverse environmental conditions. Then, we develop a Systematic Literature Review (SLR). Within the corpus of examined works, we have rigorously assessed the models' structure, types and quantity of gestures recognized by these models, the dataset construction, and the sample size of the datasets. We have also analyzed the sensor technology, the model accuracy, and the computational processing time associated with these models. Through this exhaustive review, we have synthesized a comprehensive understanding of the state of the art of the field of hand gesture recognition using infrared information. This work involved building a dataset using the Leap Motion Controller sensor. This dataset was created to training and testing our proposed models. Our dataset comprises an extensive collection of data points obtained from 56 participants, encompassing a diverse set of 9 static and dynamic gestures. This dataset is composed of total of 15,120 individual samples, divided into 8,400 static samples and 6,720 dynamic samples. With this dataset as a foundation, the subsequent chapters delve into model development and evaluation. Also, this thesis focuses on evaluating hand gesture recognition

models using feature selection and extraction methods, employing traditional machine learning algorithms as: ANN, SVM, kNN, and DT. Meanwhile, we evaluate models with automatic feature extraction via CNN and memory-enhanced models utilizing BiLSTM. These features are also evaluated in ANN, SVM, and Softmax classifiers. In conclusion, the model employing BiLSTM for feature extraction and evaluated with ANN attains a recognition accuracy of 95.73%. Conversely, the model utilizing CNN for feature extraction and assessment with ANN achieves a recognition accuracy of 91.67%. Notably, these accuracies significantly diverge from the automatically extracted features evaluated with ANN, which yield a recognition accuracy of 83.23%.

Keywords -

Hand Gesture Recognition, Leap Motion Controller, Feature Selection, Feature Extraction, Deep Learning

PROLOGUE

The integration of artificial intelligence into daily lives encourages the exploration of technologies that facilitate engagement with this dynamic field. The field of hand gesture recognition is an open way to communicate without words and is a fascinating challenge to researchers. Imagine a future where we can control our devices, communicate with each other, and even play games using our hands alone.

Hand gesture recognition is a complex task that requires Machine Learning to solve. This is because the human hand has many degrees of freedom and can be used to make an infinite variety of gestures. Additionally, hand gestures can vary depending on cultural background, individual style, and even the environment in which they are performed.

Despite these challenges, machine learning has significantly progressed in hand gesture recognition in recent years. This is due to many studies carried out about research methods of feature extraction and selection, as well as the development of new algorithms, using classic machine learning and deep learning, where these algorithms can learn complex patterns from data.

The output of a hand gesture recognition system has great potential to be used in a wide range of applications. For example, it could be used to control devices such as computers, smartphones, and TVs. It could also be used to communicate with people with impairments at speaking or writing. Additionally, hand gesture recognition could be used to create new and innovative games and entertainment experiences.

In this context, in this thesis we present a comprehensive analysis of the state of the art in hand gesture recognition, including feature extraction and feature selection methods. I also analyze the development of hand gesture recognition models and the behavior with both classical machine learning algorithms and deep learning algorithms.

Chapter 1

Problem Description

Contents

This chapter provides an overview of real-time hand gesture recognition models using infrared information and machine learning algorithms. We present a description of the problem, the objectives that drive the development of the study. Finally, we present the organization of this thesis.

1.1. Problem Description

Gestures are natural expressions of the body that people widely use to communicate [1]. However, hand gestures are considered to be movements performed in the form of unconventional language. Hand gestures are especially used by hearing or speech-impaired people for communication [2]–[4], including the growing demand for more natural and intuitive human-machine interfaces. In [5], it is mentioned that 93% of people use unconventional or non-verbal communication in their daily lives. Hand gesture recognition involves tracking hand movements and retrieving information describing a gesture.

In the modern world, touch-free interaction has become increasingly relevant, especially in contexts where hygiene and the reduction of touch surfaces are priorities. Interfaces

based on hand gestures offer an intuitive and efficient way to control electronic devices, virtual and augmented reality systems, and medical and rehabilitation applications. However, one of the main challenges is to achieve accurate and robust gesture recognition in various environmental conditions and with different users.

In this context, the problem of hand gesture recognition systems has been of great interest to the research community. This is not a trivial problem because it is related to the problem of pattern recognition and feature extraction. Gesture recognition consists of feeding a set of features into a classifier. Classification tasks involve assigning predefined labels or categories to input data based on its features, with algorithms learning to categorize them into predefined classes. Recognition tasks entail multiple classification processes because the recognition returns the class to which the gesture belongs and the instant in which the gesture was performed. While both tasks involve making predictions about input data, their objectives differ significantly: classification centers on label assignment, whereas recognition emphasizes pattern identification.

The metric for evaluating the recognition is shows in the following expression $\rho = 2 * \frac{|A \cap B|}{|A|+|B|}$; where A and B represent vectors of data. A is original signal and B is classified signal. Both the original signal A and the classified signal B, are aligned to determine the validity of the recognition. Also, we have defined a threshold. The threshold is 0.25 $\tau = 0.25$. This value is used to confirm the recognition. If the signals intersect, and the value of the intersection is higher than the threshold $\tau = 0.25$, the signal is accepted as recognized $\rho \geq \tau$. On the other hand, the signal will be unrecognized if the signals do not intersect. The output of a hand gesture recognition system can be used in other systems or in other fields such as: medicine [6], [7], virtual reality [8], [9], sign language communication [10], [11], human-machine interaction [12], [13], human-robot interaction, among others.

The researchers in the attempt to solve the hand gesture recognition problem have proposed the use of different types of sensors. The proposed sensors are: Gloves, RGB cameras, electromyographic, electroencephalographic, and infrared sensors. Using gloves can be uncomfortable for hand movements because it is external artifact that could affect the movements. The use of RGB cameras involves dealing with problems such as segmentation problems, finger occlusion, and illumination changes. The electromyographic and electroencephalographic sensors face problems such as noise generated by the sensors or the environment, signal variation in the electrodes due to sweating, and putting and taking off the sensors. In this context, infrared sensors are an alternative for implementing hand gesture recognition models because these devices do not have the problems described

above. We use the Leap Motion Controller (LMC) in the present project. LMC is a specialized hand-tracking device that provides spatial position, directions, and images. It is also a very accurate, inexpensive, and portable.

In addition, the use of infrared signals in combination with machine learning algorithms enables the creation of adaptive systems that can learn and improve over time, adapting to individual user variations and changing environmental conditions. This not only improves the usability and acceptance of these technologies, but also opens the door to new applications.

Researchers have proposed different models to solve the hand gesture recognition problem. A hand gesture recognition model consists of different modules: data acquisition, preprocessing, feature extraction, classification, and post-processing. In the model, the classifier could be designed using machine learning, especially when finding a mathematical or statistical model of hand gesture is very difficult or even impossible. Finding a mathematical model of hand gesture requires knowing the dynamics of the problem, and the problems are usually complex in their behavior. The statistical model of hand gesture must take into account all the variables involved in the problem and what is the behavior.

In the context of hand gesture recognition using the LMC, the problems encountered in the scientific literature are the following:

1. Lack of public datasets containing spatial positions, directions, velocities, and images
2. Lack of protocols for data acquisition with the LMC
3. Lack of recognition algorithms, only classification algorithms are reported
4. Decrease in model performance due to finger occlusion
5. Decrease in model performance due to excessive feature extraction functions

The problems to be solved in this work are 1, 3, and 5. Figure 1.1 shows the scheme of how the thesis is developed to cover the problems encountered. In relation to problem 1, a dataset will be created because the datasets presented in the scientific literature are generated for specific problems. In addition, the existing datasets are generated with very few users, which could lead to a problem of lack of generalization of the algorithms. Problem 3 is studied because many authors report the classification problem as a recognition problem. Finally, problem 5 is addressed because feature selection and feature extraction are directly related to the accuracy of the recognition models. In addition, the hand gesture recognition problem treated with machine learning algorithms works in a scenario very close to overfit-

ting. This is because these kinds of problems have a high dimensionality. In this sense, it is necessary to address the problem of feature selection and extraction.

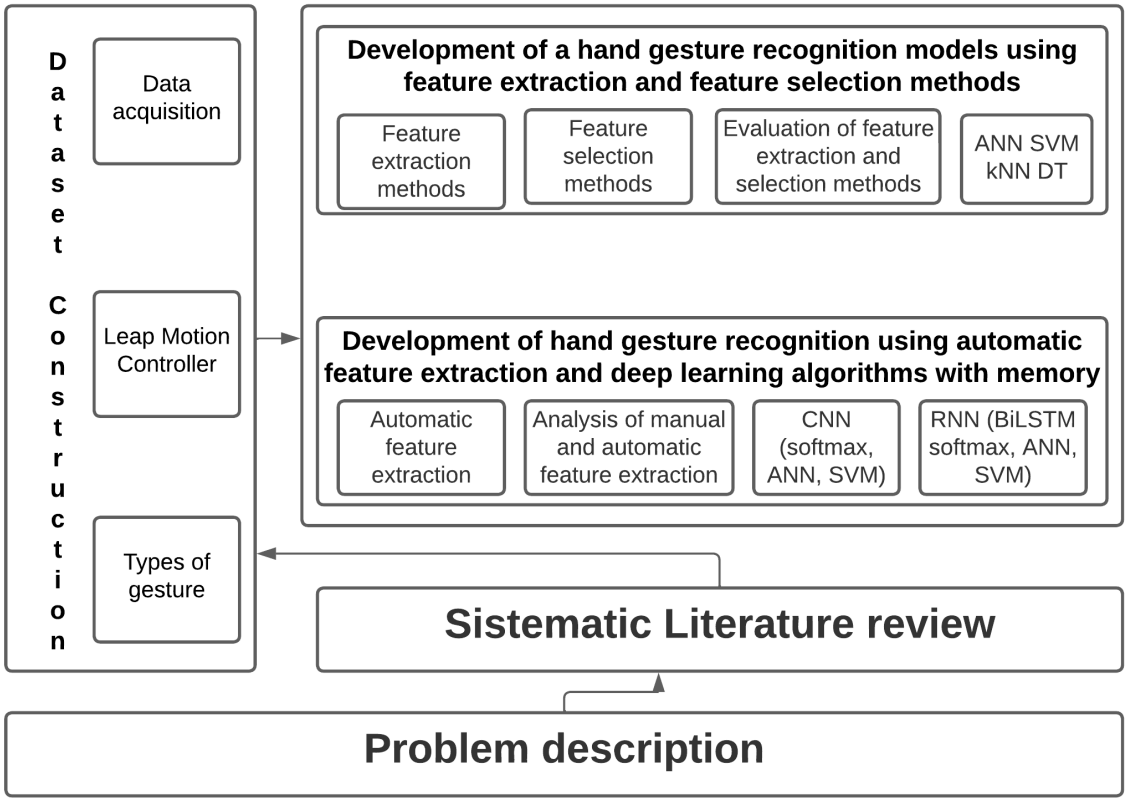

Figure 1.1: Scheme for the development of this thesis

1.2. Objectives

1.2.1. General Objective

Develop hand gesture recognition models using machine learning and information acquired with the LMC to achieve high classification accuracy and real-time recognition.

1.2.2. Specific objectives

- Investigate how the number of features affects hand gesture recognition models' classification and recognition accuracy and processing time.
- Investigate how parameters and hyper-parameters of classification algorithms affect

classification and recognition accuracy.

- Investigate the architecture of hand gesture recognition models based on parametric, non-parametric, and deep learning algorithms.

1.2.3. Contribution of the thesis

1. Generation of a dataset of signals acquired with the LMC, using 56 users, each user repeating each gesture 5 times, for a total of 9 gestures: 5 static and 4 dynamic gestures.

2. Systematic literature review on hand gesture recognition models using machine learning algorithms and infrared information.

3. Evaluation of manual feature selection and feature extraction methods to achieve high accuracy on the hand gesture recognition model using infrared signals and machine learning algorithms.

4. Evaluation of the classifier that best matches the manual features obtained in the previous step and achieves high accuracy on the hand gesture recognition model using infrared signals and machine learning algorithms with real-time processing.

5. Evaluation of automatic feature extraction techniques in advancing the performance and time of processing of hand gesture recognition systems.

6. Evaluation of models employing both manual and automatic feature extraction techniques to determine the most suitable approach for addressing the problems of hand gesture recognition using real-time infrared signals.

1.3. Document description

In this thesis, we developed a comprehensive systematic literature review (SLR) that is presented in chapter 2. This SLR analyzes several HGR models proposed in the scientific literature. The topics analyzed include the structure of the models, the types and number of gestures, the construction of the datasets, the number of samples obtained, the type of sensors used, the accuracy of the models, and the processing time. In chapter 3, we present a dataset where the data were acquired using the LMC sensor. The dataset comprises sam-

ples from 56 subjects, consisting of 9 static and dynamic gestures, with a total of 15,120 acquired samples, including 8,400 static and 6,720 dynamic samples.

In addition, in chapter 4, we present the evaluation of a hand gesture recognition model using feature extraction and feature selection methods.This chapter evaluates methods of feature selection as: Maximum Relevance and Minimum Redundancy (MRMR), Sequential, Neighbor Component Analysis without Parameters (NCAsp), Neighbor Component Analysis with Parameters (NCAp), Relief-F, and Decision Tree (DT), using traditional classifiers as artificial neural networks (ANN), support vector machine (SVM), k near neighbors (KNN), and decision trees (DT).

Chapter 5 evaluates hand gesture recognition models incorporating automatic feature extraction together advanced deep learning algorithms. This chapter critically examines the performance of CNN and BiLSTM networks when coupled with Softmax, ANN, and SVM in the final layer. By rigorously assessing the behavior and efficacy of these architectures in the context of gesture recognition, valuable insights are gleaned into their respective strengths and weaknesses. Furthermore, in this chapter we compare the results obtained from the evaluation of manual feature extraction in chapter 4 and the results obtained from the evaluation of automatic feature extraction in HGR model. Finally, in chapter 6, presents the conclusions.

Chapter 2

Systematic literature review

Contents

2.1. Abstract

This chapter presents a systematic literature review on hand gesture recognition (HGR) using machine learning and infrared information. The SLR is used to place the problem in its

proper context within the scientific literature. In this sense, this SLR is performed to verify the state of the art of existing HGR models. In addition, this SLR is performed using the Kitchenham methodology. This study reviews the architecture of the proposed HGR models in the scientific literature, also analyzes the types and quantity of gestures used for developing the models. It also analyzes the protocols for the acquisition and evaluation of data, the types of sensors used, and the datasets used for model development. Also, this chapter shows the types of learning that have been used to train HGR models, the processing time, and the accuracy of HGR models. We also analyze the methods of preprocessing, feature extraction techniques, and classification algorithms used. Finally, based on the systematic literature review, we present a list of gaps found and describe the problems that are investigated in this thesis.

2.2. Literature review of hand gesture recognition using machine learning and infrared information

Corporal or facial movements are considered a type of gesture used to communicate with other people [14]. Hand gestures are widely used, especially in non-verbal communication. These types of gestures transmit emotion and feelings or execute tasks in human-machine environments. This has resulted in a field of high interest to researchers, especially in the computer science area.

Hand gesture recognition (HGR) is not a trivial problem. This consists of identifying the type of gesture executed and the time when the gesture is performed. In this sense, HGR is considered a problem of feature extraction and pattern recognition [15]. The response of an HGR system can be used as an input signal for the operation of systems such as: Human-computer interaction [16], [17], robotics [18], sign language translation [19]–[22], virtual reality [23], [24], augmented reality [25], medicine [26], [27], among others.

The an SLR is considered a secondary study, which systematically reviews primary studies related to research questions. An SLR aims to recover information that answers the research questions, leaving the possibility of adding scientific evidence [28]. Additionally, it allows the researchers to know how a problem has evolved and the depth with which it has been investigated.

For developing this SLR, we use the Kitchenham methodology. This methodology presents three phases: planning, conducting, and reporting. Figure 2.1 illustrates the phases and

subphases recommended by the Kitchenham methodology.

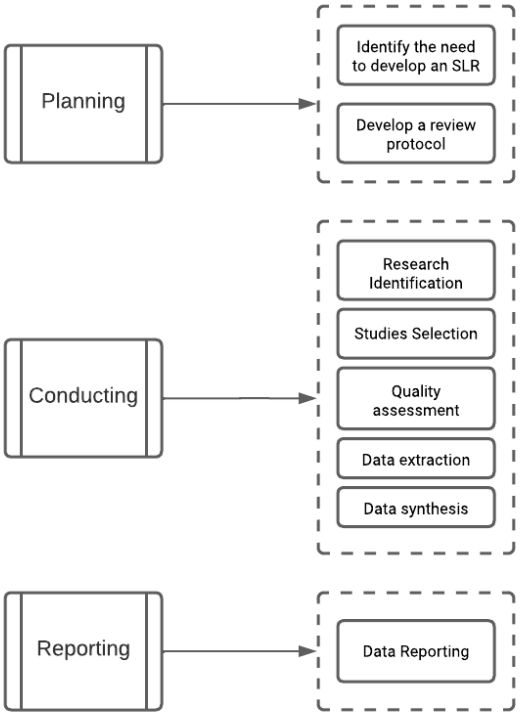

Figure 2.1: Phases of the Kitchenham methodology

2.2.1. Planning phase

In this phase, we reviewed secondary studies related to the problem and defined the protocol for performing the SLR. A secondary study involves the analysis and synthesis of existing data or literature to address specific research questions or objectives. These studies play a crucial role in aggregating knowledge, identifying gaps or inconsistencies in the literature, and generating new insights for further investigation.

Identification the need to develop an SLR

We developed a comprehensive search for secondary works on the hand gesture recognition problem using machine learning and infrared information. The search for secondary sources aimed to identify compilations and commentaries related to primary empirical research on hand gesture recognition using machine learning and infrared data. This attempt serves both to address research questions and to provide guidance for further investigation. In this study, we used specific search strings that included the following terms: systematic literature

review, state of the art, review, survey, hand gesture recognition, tracking, machine learning and infrared. The selection criteria for the papers reviewed in this section include papers published in reputable journals and conferences, and retrieved from the following databases: IEEEXplorer, ACM Digital Library, Willey Online Library, Science Direct, and Springer.

The following search string was used: (((hand AND gesture AND (recognition OR tracking)) AND "machine learning") AND (infrared OR ("infrared information")) AND (("systematic literature review") OR ("state of the art") OR review OR survey). The search string employed encompasses all topics pertinent to the subject of the study as delineated in the preceding paragraph, with the aim of ensuring inclusivity and comprehensiveness, thereby minimizing the inadvertent exclusion of any relevant works. Nevertheless, despite our thorough investigation, no specific literature reviews focusing on Hand Gesture Recognition with machine learning and infrared data were identified at the time of this study. However, seven related papers were reviewed in the context of hand gesture recognition using machine learning and computer vision algorithms. [29]–[35] present studies of models based on both colour and depth images. In addition, the authors in [30], [36]–[38] highlight the challenge of hardware limitations and processing time in machine learning models that operate in real time. [39].

Development review protocol

The definition of the protocol is a guide that helps to address the problem and identify the primary studies that will be part of the SLR. For the development of this work, the protocol is based on research questions, strategies for selecting primary studies, selecting relevant studies, assessing the quality of the selected studies, and discussing the results.

The definition of the research questions is the most critical step, since the retrieval of evidence from the primary studies is based on these questions. In this sense, we define a generic model of machine learning, population, intervention, and outcomes to describe the research questions that will guide this work. The generic machine learning model consists of data acquisition, preprocessing, feature extraction, classification, and postprocessing modules. Figure 2.2 illustrates the generic machine-learning model. The population will be all instances of the gestures generated by the hand. The gestures are defined as open hand, close hand, wave in, wave out, and pinch. The intervention will be the models based on machine learning and infrared information. Finally, the outcomes will be recognition accuracy and processing speed. In this context, the research questions which guide this work are:

General research question

- What is the state of the art for existing hand gesture recognition models that use machine learning and infrared information?

Specific research questions

- What is the architecture of the proposed models for hand gesture recognition based on machine learning and infrared information?
- What are the protocols, types of sensors, and types of dataset used to develop hand gesture recognition models based on machine learning and infrared information?
- What types of learning (supervised, semi-supervised, unsupervised, or reinforcement learning) have been used to train hand gesture recognition models with infrared information?
- What are the processing time and recognition accuracy of hand gesture recognition models that use machine learning and infrared information?

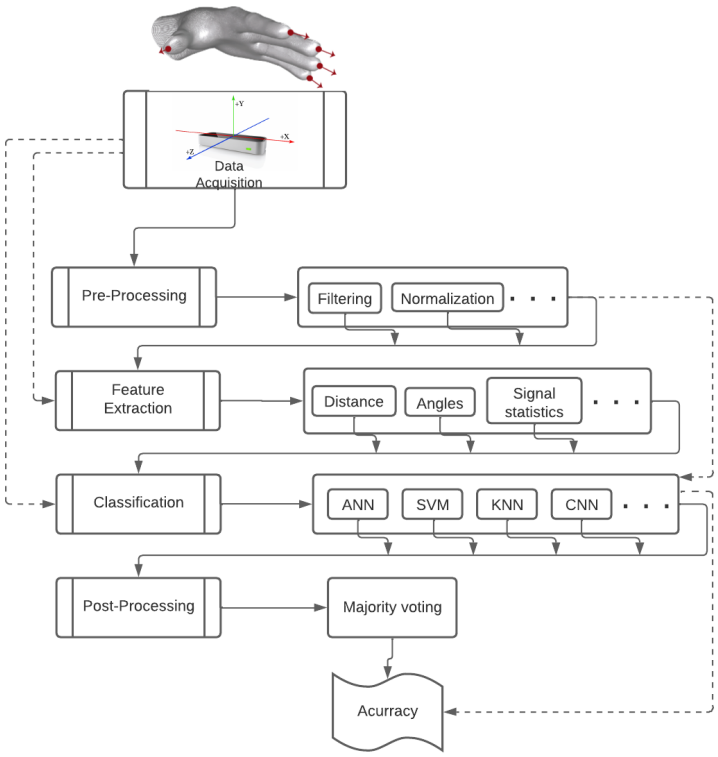

Figure 2.2: Generic model of a HGR algorithm where solid lines mean continuity of processes and dashed lines mean that processes that can be executed by skipping other processes

Strategy for searching primary studies

This section defines the scientific databases used to search for primary studies related to real-time hand gesture recognition using machine learning and infrared information. The databases are defined in Table 2.1. Scientific databases are indispensable tools, providing a rich repository of structured information critical to research and analysis. These databases provide access to diverse peer-reviewed journals, scholarly articles, and experimental data across many disciplines. These repositories also enable researchers to explore existing knowledge, validate hypotheses, and derive insights to further their investigations. In addition, scientific databases often facilitate collaboration and knowledge sharing within the research community.

Databases	URLs
ACM	http://dl.acm.org/
IEEE Xplorer	http://ieeexplore.ieee.org/
Science Direct	http://sciencedirect.com/
Springer	http://link.springer.com/
Willey	http://onlinelibrary.wiley.com/
ArXiv	arxiv.org

Table 2.1: Databases used for searching primary studies for this SLR

In addition, we define keywords that cover a wide range of research questions. These keywords are used to develop search strings. The general search string used is:

((((((hand gesture) OR (hand poses)) AND recognition) OR (hand tracking)) AND machine learning) AND ((infrared) OR (infrared information) OR (Leap Motion) OR (Kinect))).

Selection of relevant studies

In this phase, we apply the following steps to select the primary studies for the SLR:

1. Select HGR articles only defined in the databases described in Table 2.1.
2. Constrain the search of articles between 2015 and 2019 and articles of conferences and journals.
3. Select all studies where the title contains any of the following phrases:
   (a) Hand gesture recognition
   (b) Hand poses
   (c) Hand tracking
   (d) (a),(b), or (d) with any machine learning algorithms
   (e) (a),(b), or (d) with any infrared device
4. If the title does not contain any of the topics in point three, we search in the abstract, keywords, and conclusions whether the article mentions infrared sensors, machine learning algorithms, or hand gesture recognition. If this condition is met, then the article is selected.

5. The selected articles are registered in a spreadsheet. The features retrieved from each article are types of gestures, types of input, data origin, types of subjects, feature extraction, classifiers, number of samples, reported measures, article origin, and methodology.

6. Applying exclusion and inclusion criteria.

7. Register the chosen articles, after applying step 6 in a second spreadsheet. If this condition is met, then the article is selected.

8. Apply the Likert scale to rate the selected articles and determine scientific validation.

In addition, we have defined variables that allow us to obtain information to answer the research questions posed in this SLR. These variables are: Model structure, data set construction, training parameters, processing time and speed, and accuracy.

2.2.2. Conducting phase

In this phase, the primary studies related to the problem are identified, and according to the evaluation metrics, it is decided which ones will become part of the SLR. From the selected studies, we extracted all the evaluation metrics and the relevant information such as model structure, data set construction, training parameters, time of processing, speed of processing, and accuracy to help answer the research questions.

Research identification

In this subsection, we applied the search strings in scientific databases, then selected only articles that contain in the title HGR, hand poses and hand tracking. Also, these 3 topics are related to machine learning algorithms and infrared devices. If these topics are not in the title, we searched in the abstract, keywords and conclusions. In addition, if the articles mention infrared sensors, machine learning algorithms, or hand gesture recognition and are published in congresses or journals between 2015 and 2019. Then, the most relevant information from these studies is extracted and stored in spreadsheets.

This is the first step for answering the research questions. In this process, extracting as much information as possible about the research problem is necessary. In this sense, we obtained 1174 papers from indexed databases and 231 papers from non-indexed databases.

Studies selection

This subsection defines the inclusion and exclusion criteria. The articles selected in the previous subsection will be assessed according to these criteria. This evaluation determines the primary studies that meet the conditions and are passed on to the following step. The inclusion and exclusion criteria used for this SLR presented in Table 2.2.

Inclusion	- Articles that use machine learning and infrared information for hand gesture recognition. - Only articles from databases shown in Table 2.1. - Only articles from congresses and journals. - Peer-reviewed items. - Works that present models or that compare models. - Publications between January 2015 and December 2020
Exclusion	- Articles that are not related to hand gesture recognition. - Articles that do not use infrared information and machine learning for hand gesture recognition. - Articles with years of publication earlier than 2015. - If the publications do not define population, intervention, and outcomes (accuracy and time). - All articles that are not in English. - Works that present only applications and do not propose a model.

Table 2.2: Criteria for inclusion or exclusion of studies in the SLR

Applying the inclusion and exclusion criteria yielded 203 articles. Additionally, we exclude articles that do not present models, population, intervention, and outcomes, obtaining 69 articles.

Quality assessment

This doctoral thesis employs a rigorous methodology for conducting an SLR, emphasizing quality evaluation criteria. These criteria are defined and discussed with a peer reviewer, in this case, the appointed tutor. Selected studies undergo evaluation based on these criteria to ensure the credibility and reliability of the SLR findings. To avoid researcher bias, discus-

sions with the peer reviewer are integral. Table 2.3 presents the quality evaluation criteria for transparency and reproducibility, enhancing the trustworthiness of the SLR outcomes.

Quality criteria	**a**	**b**	**c**	**d**	**e**
The findings are credible				0.5	
The findings are important				0.5	
The research brings new knowledge			0.25		
The evaluation address well its original aims and proposal				0.5	
The scope of research let new researches				0.5	
The basis for evaluating the result is clear				0.5	
The research design is defensible			0.25		
The sample design, target selection of classes, is well document		-0.5			
The data collection was well carried out		-0.5			
The approach, formulation and, analysis of the problem has been adequately carried out				0.5	
The diversity of perspective and context has been explored (related work)				0.5	
The links between data, interpretation, and conclusions clear			0.25		
The reporting is clear and coherent			0.25		
The theoretical contributions, the perspectives, the values that the research leaves are clear				0.5	
The research process has been adequately documented				0.5	

Table 2.3: Assessment quality criteria of the articles using a Likert scale weighting. This is an example of how to proceed with the evaluation of the articles.

The weighting criteria are: a) strongly disagree, b) disagree, c) neither agree nor disagree, d) agree, and e) strongly agree. The researcher evaluates each quality assessment criteria reported in the papers, giving a weighting of -1, -0.5, 0.25, 0.5, and 1 respectively. We chose these weights based on the principle that when equivalent evaluation criteria exist in both sets (a and b) and (d and e), the sum equals zero $\sum a + b + d + e = 0$. This criterion ensures that the analyzed articles are positioned in the middle of the quality spectrum, with a summation of $\sum c$ thus maintaining a balanced assessment framework.

In the case that the summation of the weights a,b,d,e sets are not zero, $\sum a+b+d+e \neq 0$, the summation of the set c is taken as the threshold $\tau = \sum c$, and for an article to be part of the review, the value of the total summation of the weights must be equal to or greater than the threshold $q_s = \sum a+b+d+e \geq \tau$. After this evaluation 44 articles are included in the SLR. 12 from IEEE, 5 from ACM, 15 from ScienceDirect, 11 from Springer, and 1 from Willey.

Data extraction

In this section, we retrieve information about the selected articles that will help answer the research questions. We retrieve information about: Model structure, dataset construction, training parameters, processing time and speed, and accuracy of the reported models. To avoid bias, the retrieved information is peer-reviewed.

In [25] it is presented a dynamic gesture recognition system integrating LSTM and CNN networks to evaluate six classes of gestures. Data collection involves 3D spatial positions and finger velocity captured using a Leap Motion Controller (LMC). Single-take records of gestures are manually segmented with video assistance, and an LSTM-based algorithm is proposed for automated labeling. Post-processing consolidates frames with short gaps between gestures. The system comprises recognition and classification modules, utilizing LSTM and CNN architectures. Model training employs cross-validation, yielding 98.4% accuracy and 125ms response time. However, details on data acquisition protocols and contributors are lacking.

The paper [40] introduces a real-time gesture recognition system employing the Kinect sensor, capable of recognizing 16 gestures. Depth images are utilized to address various challenges such as background interference and illumination changes. Preprocessing techniques involve calculating pixel differences to accommodate subject mobility and employing the footfill algorithm to define hand regions. Feature extraction is conducted using SIFT and SURF techniques. Model training employs SVM with linear and radial kernels on an 8000-image dataset, yielding accuracies of 98% with SURF and 91% with SIFT. Average processing times are 0.12s with SURF and 0.30s with SIFT. While a data acquisition protocol is briefly detailed, the number of subjects contributing to the dataset remains unspecified.

In [41] it is presented the development of a rehabilitation platform, also contributing to a systematic literature review through its classification and recognition model for eight static and dynamic gestures. The dataset comprises data from 30 injury-free subjects, each re-

peating gestures 100 times, collected using a Leap Motion Controller (LMC) at 150 frames/s. Preprocessing involves smoothing, while feature extraction includes window division (w = 20) and computation of mean, distance, and angles between fingertip and palm center. Static gestures are classified using discriminant analysis (DA) and SVM, while dynamic gestures utilize hidden Markov models (HMM), achieving accuracies of 99.09% and 98.76%, respectively. The rehabilitation system evaluation reports an accuracy of 80%, though processing time remains unreported.

In [42] a model for recognizing 28 Arabic alphabet letters is introduced, capable of detecting both static and dynamic gestures. Data acquisition employs the Leap Motion Controller (LMC) and the Kinect, capturing velocity, orientation, and depth images. To ensure dataset consistency, data from both sensors are standardized to millimeters. Preprocessing involves principal component analysis (PCA) for length adjustment and redundancy elimination. Feature vectors are constructed from normalized finger lengths and angles, feeding into an SVM classifier with a Gaussian kernel. Model training with 1121 samples and 280 validation samples achieves 93% training accuracy and 86% testing accuracy. However, details on post-processing and processing time are omitted.

In [43], the authors propose a system for recognizing sign language and semaphoric hand gestures using 30 gestures (18 static and 12 dynamic). Spatial coordinate data is gathered through CML, focusing on joint angles for various fingers. They train and test the model on the SHREC database and introduce a proprietary dataset with 1200 samples from 20 subjects aged 20 to 28 (15 males, 5 females). The data acquisition protocol involves a 5-second sample at 200 Hz. The dataset lacks preprocessing, but a feature vector includes angles between phalanges, particularly for the thumb, along with spatial fingertip positions. These features feed a classifier named DLSTM, a combination of RNN and LSTM. The model achieves an accuracy of 96.4102%, precision of 96.6434%, and recall of 96.4102%. However, the authors do not provide a detailed protocol for reporting data.

In paper [44] a gesture recognition system is presented with a focus on recognizing Arabic numerals (0 - 9) and the capital letters A and Z through 12 dynamic gestures. The study aims to showcase the efficiency of data capture using the Leap Motion Controller and proposes an effective gesture recognition method based on finger positions and hand orientation. The authors construct a dataset with 12 subjects, each repeating the 12 gestures 10 times, resulting in 1200 samples. While the paper lacks a preprocessing module, it emphasizes the deterministic learning theory and employs the Gaussian function as a radial basis function. The system is divided into a training phase and a recognition phase. During

training, 3D finger data acquisition, calculation of fingertip motion angles, feature selection, and modeling of dynamic finger motion are performed. In the recognition phase, new data acquisition, fingertip motion angle calculation, feature selection, and the construction of a bank of dynamic estimators occur based on the training results. Feature selection includes spatial positions and angles, feeding a neural network with a radial basis function. Cross-validation is employed for training, yielding reported accuracies of 95.83% with 2 folds and 97.25% with 10 folds. The paper, however, does not provide values for classification and recognition metrics.

In [45] a medical image manipulation system is presented, designed for sterile environments and employing 11 dynamic gestures identified through discussions with surgeons during technical hospital visits. The Leap Motion Controller (LMC) captures spatial positions and finger/palm directions to create a dataset of 550 samples from 10 individuals. The model encompasses data acquisition, feature extraction, and classification modules. While the paper doesn't explicitly mention a preprocessing module, data normalization within the range [-1, 1] is applied. The feature vector is constructed using the window splitting technique (w = 20), focusing on palm center spatial positions and fingertips. Extracting arithmetic mean, standard deviation, covariance, and root mean square results in six vectors, which are concatenated to form the feature vector. These features feed into an SVM classifier with nonlinear Gaussian radial basis functions as the kernel. Model training employs cross-validation, yielding an accuracy of 81%.

In [46], the authors tackle the finger occlusion issue using two Leap Motion Controller (LMC) sensors. They employ three gestures for experimental demonstration, focusing on estimating fingertips' position, palm characteristics, normal and direction vectors, and hand rotation. To enhance accuracy, an offline classifier is trained using an artificial hand, capturing data from the two LMCs at a sampling rate of 120 frames/sec, resulting in a dataset of 108 samples. Instead of depth image analysis, the authors directly analyze hand position. Calibration of sensors at a 150-degree angle is crucial due to potential field-of-view overlap. Preprocessing involves transforming data from a global to a local hand coordinate system. The feature vector includes x, y, and z components derived from palm position, normal plane, direction values, rotation at the Roll angle, dot product between normal vector and direction, and a sensor confidence estimate. These features feed into an SVM classifier. While details on model training are lacking, a recognition accuracy of 90.80% is reported. However, the paper does not specify whether this accuracy pertains to classification or recognition, and no protocol for the reported values is provided.

The extraction of information from the papers that are part of this SLR can be found in detail in the article [39]. However, in this thesis 8 articles are mentioned so that the reader can observe the data extraction methodology.

Data synthesis

In this section, we answer the research questions. The variable that contributes to research question 1 is the model structure. The model structure includes a) data acquisition, b) preprocessing, c) feature extraction, d) classification, and e) postprocessing modules. The architecture of a machine learning model is the union of all the modules or the union of some of them. The complexity of these architectures is elucidated through Figure 2.3, which presents a histogram detailing the prevalence of different module combinations across a set of articles analyzed in the SLR.

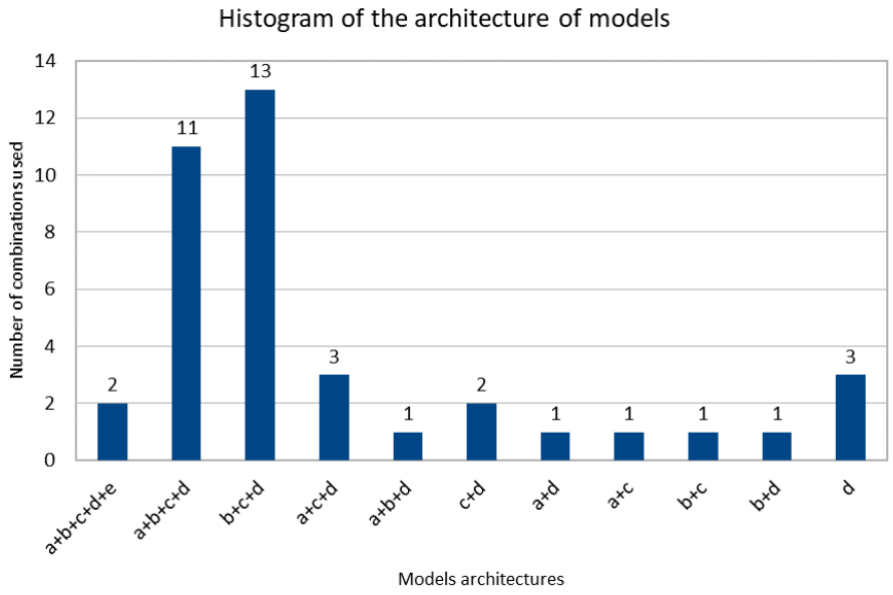

Figure 2.3: Architecture of the machine learning models for the HGR found in the SLR

In the architecture of machine learning models, researchers deploy diverse methods and techniques within each module to maximize the likelihood of achieving great accuracy. These modules are foundational building blocks, each tailored to address specific tasks or challenges inherent in the learning process. For example, in the data preprocessing module, techniques such as normalization [26], [36], [47]–[49], segmentation [50]–[53], filters [54]–[56], and feature engineering are employed to enhance data quality and relevance.

Similarly, within the feature extraction module, methods like principal component analysis (PCA) [57]–[59], convolutional neural networks (CNNs) [26], [60]–[62], or recurrent neural networks (RNNs) [26], [60], [62] may be utilized to extract meaningful patterns and representations from raw input data. The choice of algorithms and models within the learning module itself—ranging from classical methods like support vector machines (SVMs) [54], [57], [63], artificial neural network (ANN) [64], [65] and hidden Markov model (HMM) [54] to sophisticated deep learning architectures—depends on the complexity of the task and the nature of the data. Furthermore, researchers often incorporate ensemble methods, regularization techniques, and hyperparameter tuning to fine-tune model performance and mitigate overfitting.

The variable that contributes to answer the second research question is the construction of the dataset. To build a dataset, it is necessary to identify the types of sensors used. It is also necessary to identify whether the samples were collected using a data acquisition protocol. The sensors reported in SLR are Leap Motion Controller, Kinect, Intel RealSence, and Interactive Gesture Camera. Table 2.4 shows the articles that used LMC to construct the dataset and the number of samples used to construct the dataset.

Article	**number of samples**
[60]	1005
[66]	1200
[67]	550
[46]	108
[68]	2000
[69]	5000
[70]	100
[65]	3600
[71]	220

Table 2.4: Articles in which datasets are constructed using LMC and number of samples obtained.

Table 2.5 also shows the articles that used Kinect and the number of samples used to build the dataset.

Article	number of samples
[63]	8000
[72]	1200
[50]	800
[73]	3000
[58]	50000
[74]	5040
[75]	507000
[76]	1600
[77]	3280
[56]	12000
[78]	700

Table 2.5: Articles in which datasets are constructed using Kinect and number of samples obtained.

The training algorithms contribute to answering the third research question. In this sense, we analyze the machine learning types used, such as supervised, semi-supervised, unsupervised, and reinforcement learning. The SLR showed that the problem of HGR in all papers reviewed used supervised learning. We also examine the hyper-parameters for tuning these algorithms and whether these hyper-parameters are adjusted manually or automatically. In the data acquisition module the hyper-parameter adjustment is performed in a heuristic way by trial and error. The pre-processing module adjusts the hyper-parameters using the heuristic trial and error technique. In the feature extraction module, the hyper-parameter setting is done by trial and error, but the hyper-parameters can also be adjusted automatically. The algorithms used for classification perform the work automatically. It consists of mapping the input data with their respective labels, and the trained model can return a label for a new data set.

Finally, the processing time and speed, and the accuracy of the reported models are the variables used to answer the fourth research question. Both processing time and processing speed are considered critical variables in machine learning models due to the complexity and many mathematical calculations. In this context, obtaining values representing high classification and recognition rates is challenging. In addition, many models report classification accuracy as recognition accuracy.

The SLR of HGR indicates that the Support Vector Machine (SVM) algorithm is the most

prevalent choice for classification tasks. SVMs are favored for their ability to perform both linear and non-linear classification. Consequently, SVMs can compute an optimal hyperplane for classification [37], [68], [79]. On the other hand, Recurrent Neural Networks (RNNs) represent a family of classifiers adept at processing sequential data by retaining memory information within hidden layers. The review highlights RNNs' utility in classifying dynamic gestures [35] Also, the k-Nearest Neighbors (kNN) algorithm, a non-parametric approach, relies on the quantity of available data and assumes similarity between nearby features. Proximity is measured using various distance metrics. The SLR showed that the highest accuracy was achieved with the kNN [47].

2.2.3. Reporting phase

In this section, we report and discuss the findings of this SLR. In this context, one of the biggest challenges for researchers starting to work with machine learning is to obtain a dataset that represents the problem they are investigating. The techniques used in the preprocessing modules for position and spatial data are: normalization and the simultaneous use of low-pass and high-pass filters, while for images it is segmentation.

The feature extraction module is considered very important in machine learning models. The techniques reported are dimensionality reduction using principal component analysis [57]–[59], the distance between the tips of adjacent fingers, the angle between the first finger and each of the other fingers, the distance between the fingertip and the center of the palm, the angle between adjacent fingers [32], [46], [51], [54], [65], [68], [69], [72], [80]. Also, used measured statistics as mean of velocity of palm [54], arithmetic mean, standard deviation, root mean square, covariance [67], and in [81] used SIFT-SURF. Some authors propose to develop automatic feature extraction using autoencoder neural networks or convolutional neural networks.

In the classification module, researchers showed the use of artificial neural networks (ANN), K-near neighbors (KNN), support vector machine (SVM), convolutional neural networks (CNN). Also, the researchers reported the use of long short-term memory (LSTM), which is a kind of cell of neural networks with memory.

2.3. Problems found in the literature review

This SLR reveals many challenges and complexities inherent within the research of HGR problem. These challenges span a spectrum of domains, ranging from methodological limitations, data availability, and gaps in knowledge. Understanding and addressing these problems are imperative for ensuring the integrity and reliability of the review findings. This section briefly describes the problems encountered in the papers reviewed during the SLR.

2.3.1. The lack of public datasets containing spatial position, direction and image data captured by the LMC

One challenge encountered in machine learning pertains to the significant volume of data required for model development. Each researcher begins constructing a dataset tailored to the specific nuances of their problem domain, encompassing factors such as types of gestures and data acquisition devices utilized. However, a recurring issue arises as many researchers opt not to share or release their meticulously curated datasets to the public domain. Consequently, this lack of publicly available datasets poses a considerable hindrance when researchers endeavor to test the generalizability and efficacy of their models on diverse datasets, thereby complicating the validation and comparison of machine learning approaches across studies.

2.3.2. The lack of protocols to acquire data with the LMC

The absence of standardized protocols for data acquisition utilizing the LMC presents a significant challenge to the reproducibility of research findings. Without publicly available datasets or clearly defined procedures, replicating studies becomes exceedingly difficult, impeding scientific progress and hindering the validation of proposed methodologies. Moreover, the lack of established protocols increases the risk of potential errors during data collection, including improper sensor placement or suboptimal hand positioning by users. These factors can significantly impact the performance of machine learning models, leading to diminished accuracy and reliability of results. Therefore, addressing the need for comprehensive data acquisition protocols is imperative for ensuring the robustness and reproducibility of research conducted with the LMC.

2.3.3. The absence of recognition algorithms

The absence of dedicated algorithms for hand gesture recognition poses a notable challenge within the field of machine learning. While numerous algorithms are adept at solving classification problems by assigning predefined labels to input data, the task of hand gesture recognition encompasses a broader scope. In this context, classification involves categorizing input gestures into predefined classes or categories based on extracted features and learned patterns. However, true recognition extends beyond the classification and necessitates a deeper understanding of the gesture's context, intent, and temporal dynamics. Recognition encompasses multiple and repeated classifications. Unlike classification, which focuses solely on assigning labels to input data, recognition involves a holistic understanding of gestures, including their spatial variations, temporal sequences, and contextual relevance. Therefore, while classification accuracy is a valuable metric for assessing algorithm performance, true recognition accuracy encompasses a more comprehensive evaluation of the algorithm's ability to accurately interpret and respond to a diverse range of hand gestures in real-world scenarios.

2.3.4. Decrease of model performance due to finger occlusion.

Occlusion presents a significant challenge in hand gesture recognition, particularly when one or more fingers overlap with others. When the sensor attempts to estimate the positions of the hand and fingers from the captured image, occluded fingers can lead to inaccuracies in the position estimation. This occurs because the sensor's ability to discern individual finger positions is compromised when they are obscured or overlapping, resulting in erroneous estimations. Consequently, occlusion significantly diminishes the performance and accuracy of the model, highlighting the need for robust techniques to mitigate the effects of occlusion in hand gesture recognition systems.

2.3.5. Decrease of model performance due to excess features.

In machine learning, the careful selection and management of features hold profound significance, influencing the performance and efficacy of the models. Features are fundamental upon which models discern patterns, relationships, and structures embedded within the data. A meticulous curation of relevant features enhances a model's capacity to capture intricate relationships and reinforces its generalization ability, facilitating robust performance

on data instances. Conversely, an excess of irrelevant or redundant features can precipitate overfitting, where the model assimilates noise or irrelevant patterns from the training data, thereby compromising its capacity to generalize effectively. Consequently, the feature selection and engineering process emerges as a critical endeavor in developing machine learning models, with far-reaching implications for their interpretability, computational efficiency, and predictive accuracy.

2.4. Problems to investigate

For the problems defined above, our investigation aims to address several critical issues identified within the literature. Specifically, we will focus on examining the construction of datasets, addressing the absence of dedicated recognition algorithms, and exploring the detrimental effects of feature overload on model performance. By probing into these challenges, our study seeks causes, implications, and potential avenues for mitigation.

2.4.1. The absence of recognition algorithms

This subsection defines the difference between the *classification* and *recognition* problems. The **classification** problem consists of giving a data vector $\mathbf{X} = [x_1, x_2, x_3, ..., x_m]$ to a model f, and the model returns the class to which the data vector belongs $y = f(\mathbf{X})$. Figure 2.4 illustrates the classification problem.

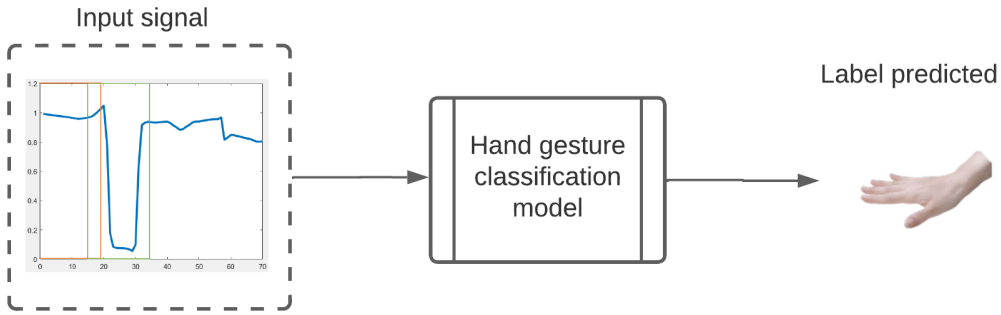

Figure 2.4: Illustration of the hand gesture classification process using a signal with spatial data.

The **recognition** problem consists of classifying the gesture to the class to which it belongs and defining its execution time. The execution time indicates the moment when the execution started and the moment when the execution of the gesture ended. The recognition

problem can be treated as a successive classification problem. And the series of successive classifications can be performed using a sliding window scheme, where the classifier returns a label and time for each window. In this sense, at the end of sliding the window through the entire signal, two arrays are obtained, an array of labels and the array of times to which the labels correspond. Figure 2.5 illustrates the recognition problem. In this context, the recognition problem is considered more complex than the classification problem.

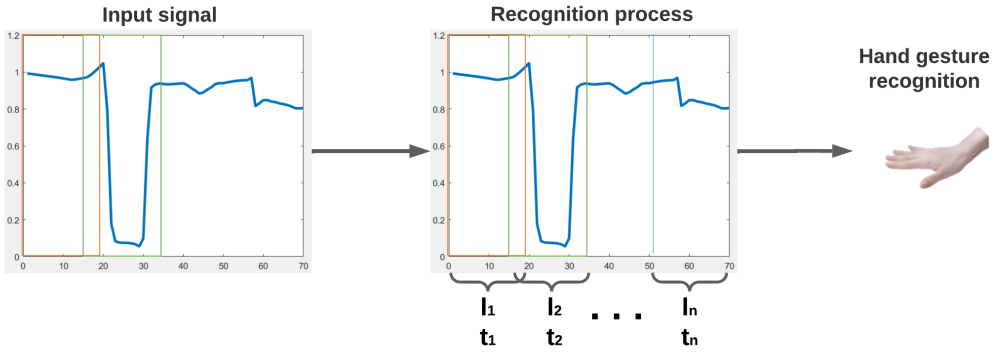

Figure 2.5: Illustration of the hand gesture recognition process using a signal with spatial data.

2.4.2. Decrease of model performance due to excess features

Working with machine learning models is challenging due to the high dimensionality and limited amount of data available for their training. This leads to the overfitting of the models. Overfitting occurs when a model remembers the training data and cannot generalize to new data. In this context, it is necessary to generate a feature extraction process. Time domain feature extraction is developed using functions that adequately represent the problem. This process's challenge is defining the appropriate combination of feature extraction functions so that the model can achieve a maximum accuracy value according to the metric used. We also use a feature selection process to define the best combination of features. Figure 2.6 shows the decrease in model performance when applying multiple feature extraction functions. It happens when the number of features is finite.

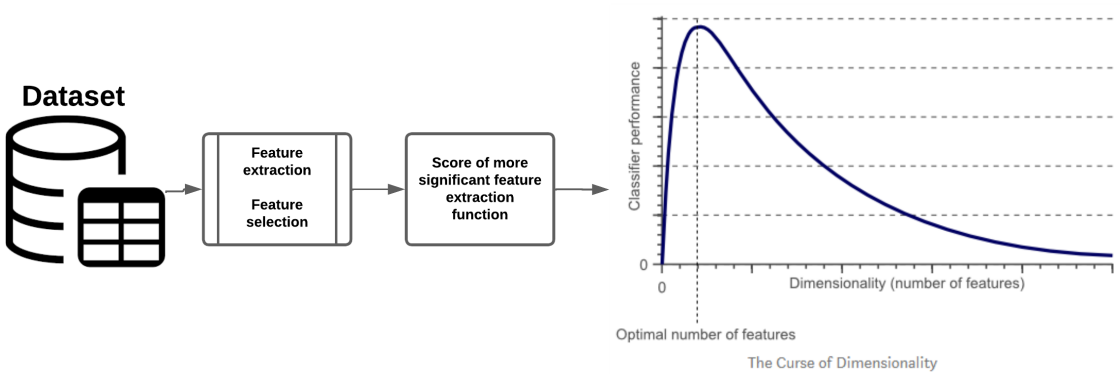

Figure 2.6: Illustration showing the selection process of the most significant feature extraction functions

2.5. Conclusions

In this chapter, we have presented an exhaustive systematic literature review concerning hand gesture recognition. To carry out the SLR, we used the Kitchenham methodology and reviewed five indexed databases: IEEE Xplorer, ACM Digital Library, Willey Online Library, Science Direct, and Springer. Also, we defined a generic machine learning model, research questions, population, intervention, and outcomes to acquire information about the problem; after the evaluation of primary works exposed in the scientific literature have included 44 articles in the SLR and concluded the following.

Concerning the first research question, we conclude that the scientific literature presents several models that try to solve the problem of hand gesture recognition using machine learning and infrared information. Mapping the generic model with the models shown in primary studies, we observed that not all works use the proposed modules. However, these models present the results of their research such as the value of accuracy of the classification and some works present the processing time.

To the second research question, we observed that the models evaluate different types and numbers of gestures, and the construction of the dataset differs in the number of samples obtained, the shape, origin, and type of sensor used. In this sense, the works are not comparable. For the third research question, we observed that the models approached the problem from the type of supervised learning. In this sense, these models present that the adjustment of parameters in the data acquisition and pre-processing modules is carried out in a heuristic way and by trial and error. In contrast, the feature extraction module is per-

formed automatically and in heuristic mode. Finally, the classification module is automatic.

Research question four is based on the accuracy and time of processing. Due to the complexity and many mathematical calculations, these variables are considered critical in machine learning models. Also, many models report the accuracy of the classification as the accuracy of recognition, which is a mistake, and not all works reported the processing time.

Finally, this chapter presents the gaps in the HGR problem, such as the deficiency in data acquisition protocols, the construction of the generic dataset that allows researchers to test their models, and measure the classification accuracy, recognition, and processing time of the algorithms. Also, there are datasets with a limited number of samples and few repetitions of the gestures acquired with Leap Motion. The lack of protocols for recognition and the lack of automatic feature extraction methods for spatial positions.

Chapter 3

Dataset construction

Contents

3.1. Abstract

A dataset is a critical part for the development of machine learning models because these models need data for training and testing. In this chapter, we present the construction of a dataset consisting of five static and four dynamic gestures. This dataset will be used for the development of this thesis study and is constructed using the Leap Motion Controller. This dataset contains data from using 56 subjects. This chapter also discusses the types of hand gestures: statics and dynamics. The interface created for data acquisition is also described. Lastly, the structure of the data from the acquired dataset is outlined.

3.2. Considerations from dataset construction

The systematic literature review presented in the previous chapter shows different datasets for the hand gesture recognition problem. These datasets could be public or private and

were designed according to the needs of the researchers. The researchers present a different number of gestures, types of gestures, number of samples, and different numbers of people forming a dataset in their datasets. In this sense, the construction of a dataset requires considerations such as: the protocol of data acquisition, the type of data, and the information of the users involved in the construction of the dataset. The dataset proposed here is distinct from those found in the SLR in terms of data volume. While the largest dataset identified in the literature contains 5000 observations, the proposed dataset includes 15120 observations. Additionally, unlike the LMC datasets in the SLR, which contain only spatial data, the proposed dataset incorporates image sequences and demographic information. Overall, the dataset supporting this thesis comprises both spatial data and images, providing a more comprehensive resource for analysis and research.

3.2.1. Data acquisition protocol

Nowadays, hand gestures are formalized to interpret the operations of some technical movements as robot movements in human-machine interaction [82], among others. To acquire data using the Leap Motion Controller, begin by placing the device on a stable surface, ensuring its LED indicator faces the individual performing the gestures. Maintain a distance of approximately 20 cm between the hand and the surface of the sensor for optimal performance. Orient the hand movements primarily along the Z-axis direction of the sensor to ensure accurate tracking. Execute the desired hand gestures within the sensor's field of view, ensuring consistency and deliberate movements to capture reliable data. Once the gestures are performed, record the data captured by the Leap Motion Controller for subsequent analysis and processing. This protocol ensures consistent and accurate data acquisition, facilitating the development and evaluation of HGR models. In addition, it allows for a certain uniformity of data among users. This process is shown in Figure 3.1.

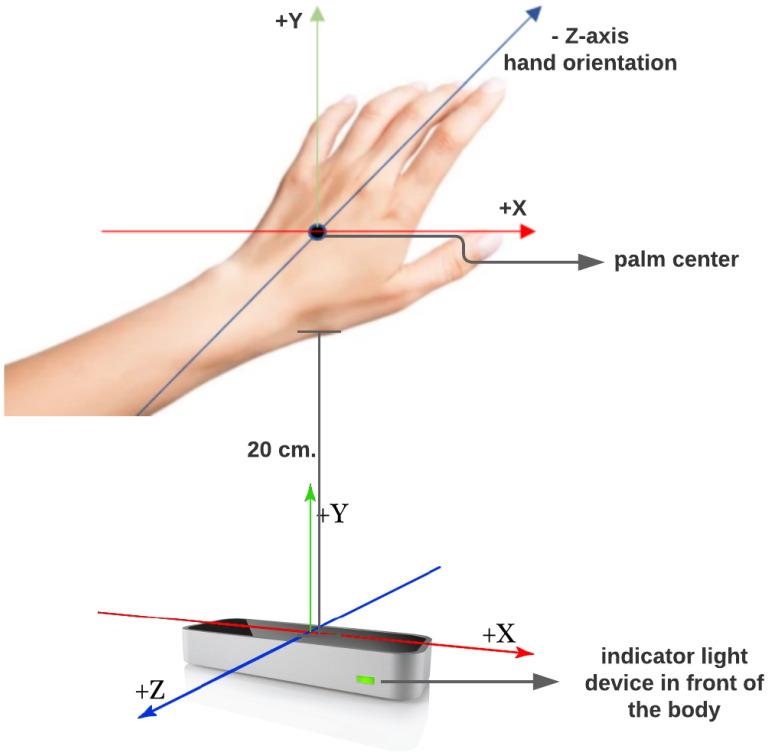

Figure 3.1: Position and orientation of the hand and of the LMC for data acquisition

In the development of this research, the sampling formula based on the normal distribution $N = \frac{Z^2 * p * q}{e^2}$; was used to ensure that the sample is representative of the target population. With Z as confidence level of 95% and a margin of error e of 13%, it was determined that a sample N of 56 users would be adequate to obtain meaningful results. This statistical approach is justified by the need to balance the accuracy of the results with the operational feasibility of data collection. By keeping the margin of error within a reasonable range, it can be guaranteed that the conclusions derived from the sample can be extrapolated to the population of men and women aged 18 to 56 years with a proportion of 90% for people aged 17 to 27 years in the central part of the country, minimizing the possibility of biases and systematic errors.

Each of the 56 users selected for experimentation represents an independent observation that contributes variability and robustness to the analysis. The consideration of a 13% error in this context allows us to capture the intrinsic heterogeneity of the individual responses and the probability distributions associated with each user. This experimental design ensures that statistical inferences and predictive models developed from the data collected are sufficiently robust and generalizable.

The dataset was built with volunteers of the Universidad Técnica de Ambato, including students and faculty members (women and men aged 18 to 46 years). The volunteers developed nine gestures, each subject repeated the gesture 30 times. The sampling period was five seconds, and the subject could perform the gesture at any time during this period. With these considerations, at least 56 users are required.

The static hand gesture dataset contains 1680 observations of each gesture, for a total of 8400 observations. The LMC has a sampling frequency of 200 Hz. However, since our dataset stores spatial position, directions, and images, the sampling frequency is reduced to 70 Hz. Each dataset instance contains data from the palm of the hand and of the five fingers, and each finger contains three **X, Y,** and **Z** channels [83].

3.2.2. Leap Motion Controller

A hand gesture recognition system uses both *invasive* and *non-invasive* devices to track hand gestures [79], [84]. **Invasive** devices are sensors embedded in the muscle of the hands and tips of fingers. Invasive sensors used for hand gesture recognition serve as tools in capturing and interpreting signals or hand movements. The invasive sensors provide real-time data, facilitating tasks such as recognizing hand gestures. These sensors may encounter accuracy issues or require regular calibration to ensure reliable performance. Moreover, prolonged use of invasive sensors, like gesture recognition sensors, may necessitate periodic adjustments to maintain functionality and mitigate risks associated with long-term implantation.

On the other hand, *two categories* of **non-invasive** sensors for HGR systems are distinguished [79]. The **first category** is sensors that cover parts of the body, such as the Myo Armband [84], [85] and Smart Gloves, which use inertial sensors, for example, accelerometers, magnetometers, and gyroscopes. These sensors improve the way of interaction. Nevertheless, they present some limitations in the sensitivity of the measurements. The **second category** is non-contact sensors, and they are based on sensors that generally use vision depth cameras that generate stereo vision. Some of these cameras are Microsoft Kinect [86], [87], Intel RealSense Camera, and Leap Motion Controller (LMC) [88]. Sensors falling into the second category prioritize user safety and comfort. However, they often encounter challenges related to sensitivity to lighting conditions, occlusion, complex backgrounds, and variations in interaction distances from the sensor [79], [89]. These sensors provide two types of data: spatial positions and images [79].

In this context, the most used sensors are the Kinect and LMC [90] due to the great demand of the market for its commercial development. The Kinect device extracts 25 spatial positions of the human body and comprises depth sensors, skeleton tracking, and color cameras [91]. The disadvantage of the Kinect is its high cost. Also, the Kinect is not a device for capturing data on a specific body part.

On the order hand, the LMC is a low-cost, accurate, and dedicated device for capturing hand movements. The LMC shown in figure 3.2. The device delivers 27 spatial positions of the hand and the fingers. Also, the LMC returns the directions of fingertips. The LMC also provides a sequence of images from 2 infrared cameras. From these images, the device estimates the spatial position of the hand and fingers.

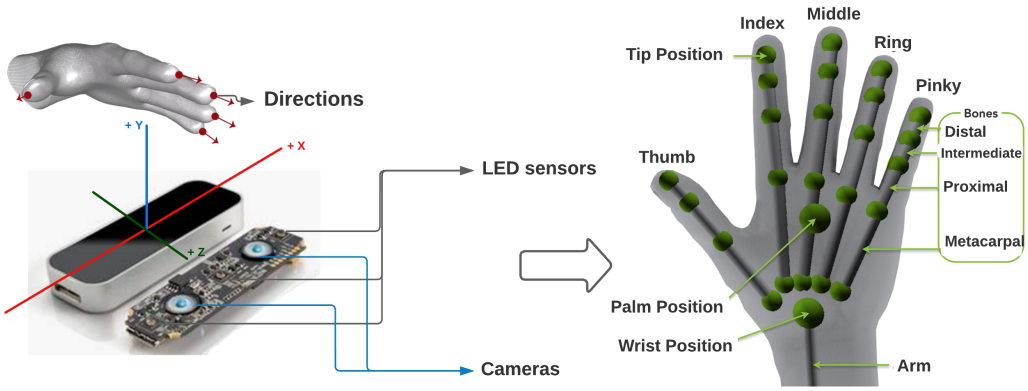

Figure 3.2: The hand features acquired with the LMC

This estimation is based on the 3D coordinate axes whose origin is in the center of the sensor. In addition, Figure 3.3 shows how the LMC can track the hand in a range of 150 degrees wide and 61 cm high, with an accuracy of 0.01 mm [92].

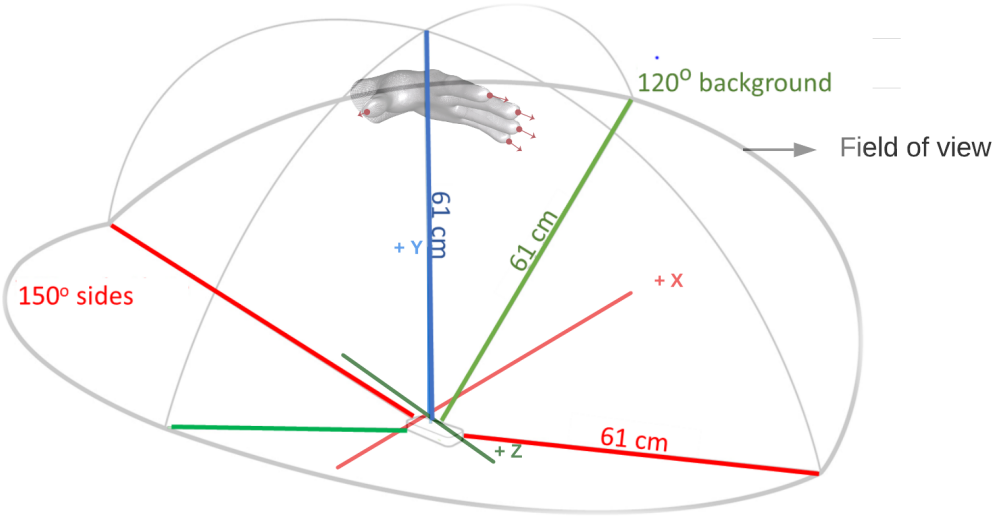

Figure 3.3: Architecture of LMC, field of view, and hand performing the gesture

Figure 3.4 shows how the LMC returns a sequence of grayscale images $f(h, w, 1), \ldots, f(h, w, T)$, where the image $f(h, w, t)$ contains a snapshot of the hand movement at time t, with $t = 1, 2, 3, \ldots, T$. Notably, such sequences can represent various gestures, as exemplified by Figure 3.4, which specifically illustrates a close hand gesture.

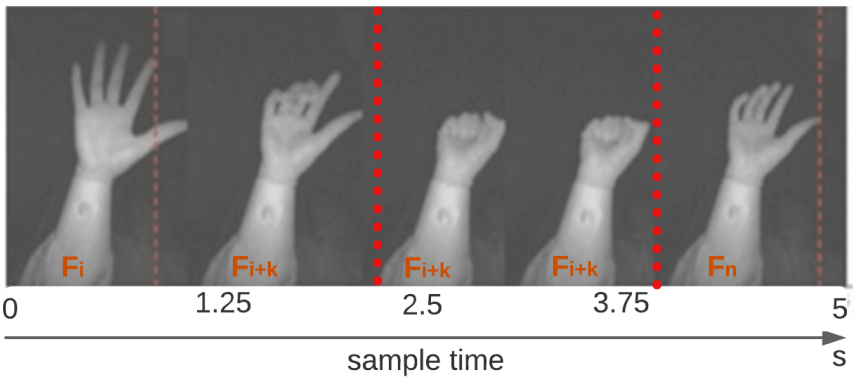

Figure 3.4: Sequence of frames that represent a hand gesture

The LMC also returns the spatial positions of the palm of the hand and the fingertips at time t. The spatial data are represented by the matrix, as shown in the following expression. $\mathbf{P}_t = [p^{(x)}_{(1,t)}, p^{(y)}_{(1,t)}, p^{(z)}_{(1,t)}, ..., p^{(x)}_{(5,t)}, p^{(y)}_{(5,t)}, p^{(z)}_{(5,t)}]^{(leap)}_t$, where $[p^{(x)}_{(i,t)}, p^{(y)}_{(i,t)}, p^{(z)}_{(i,t)}]$ is the vector with the spatial positions of the ith finger concerning the sensor coordinate axes. The LMC returns the directions of the fingertips at time t.These data are represented in the matrix $\mathbf{D}_t = [d^{(x)}_{(1,t)}, d^{(y)}_{(1,t)}, d^{(z)}_{(1,t)}, ..., d^{(x)}_{(5,t)}, d^{(y)}_{(5,t)}, d^{(z)}_{(5,t)}]^{(leap)}_t$, being $[d^{(x)}_{(i,t)}, d^{(y)}_{(i,t)}, d^{(z)}_{(i,t)}]$ the vector with the

directions of the ith finger with respect to the sensor coordinate axes.

3.2.3. Types of gesture

Gestures are divided into two types: **static** and **dynamic**. *Static* gestures refers to a hand or body posture that remains stationary or unchanged over a period of time. Static gestures involve holding a specific pose or position to convey meaning or information. The static gestures are shown in Figure 3.5, and these gestures are *Pinch, Fist, Open Hand, Wave In, and Wave Out* [93].

Figure 3.5: Static hand gestures

- **Pinch** is a gesture in which the user makes a motion in which the middle and ring fingers press together with the thumb.
- **Fist** is a movement of the hand performed by closing each of the fingers toward the center of the user's palm.
- **Open Hand** gesture involves spreading and separating all fingers, typically held in a perpendicular position to the surface or direction of reference.
- **Wave In** gesture involves aligning the little, middle, ring, and index fingers parallel to each other, generating a rotational movement of the hand around the user's internal line of sight.
- **Wave Out** gesture involves aligning the little, middle, ring, and index fingers parallel to each other, then generating a rotational movement of the hand outward, away from the user's body or central line of sight.

On the other hand, a *dynamic* gesture refers to a continuous or changing movement made by the hand or body, typically used to convey information, commands, or expressions. The evolution of these gestures must be determined in time-lapse for each one [93]. Figure

3.6 shows the dynamic hand gesture. These gestures are *swipe, circle, screen taps, and key taps*.

Figure 3.6: Dynamic Hand Gesture

- **Circle** is a movement of the index finger that rotates around its axis, forming a kind of circle in a certain number of repetitions.
- A **Swipe** hand gesture is a motion where the open hand is moved across a surface or through the air in a specific direction, typically to interact with a device or system. This gesture involves a continuous hand movement in a linear trajectory.
- **Key Taps** is a gesture in which the movement of the index finger is in the form of an arc to the hand, producing a kind of click as a pointer.
- **Screen Tap** hand gesture involves tapping one or more fingers on a touchscreen or similar interactive surface to initiate an action or select an item.

For hand gesture recognition using the LMC, a three-dimensional Cartesian plane centered at the device's origin serves as the reference. The system extracts spatial coordinates of the palm and fingertips along with the direction of the fingertips. Simultaneously, images capturing the hand movement are acquired. Additionally, demographic information of the user is gathered alongside spatial and directional data.

Utilizing spatial and directional positions for hand gesture recognition offers a comprehensive approach to capturing and interpreting hand gestures accurately. Spatial positions provide valuable information about the precise location of key hand landmarks, such as the palm and fingertips, in three-dimensional space. By capturing the spatial coordinates of these landmarks, the recognition model can discern the specific gestures being performed with high precision. Directional positions add another layer of detail to the gesture recognition process by capturing the orientation of the fingertips relative to the center of the cartesian plane. This directional data provides insights into the shape or configuration of the hand during the gesture, enhancing the model's ability to differentiate between different gestures with similar spatial positions. By combining spatial and directional information, the hand

gesture recognition model gains a robust understanding of the hand's movement in space and its orientation, allowing for more accurate and reliable recognition of a wide range of gestures.

Incorporating demographic data into the hand gesture recognition process enriches the understanding of user behavior and preferences, enhancing the overall user experience. Demographic information, such as age, gender, ethnicity, occupation, email, and whether you have suffered injuries to your right limb, provides valuable context about the user's characteristics and potential variations in hand movements based on individual traits. Age is related to factors such as motor skills development, which may influence the speed or range of motion in hand gestures. Gender differences in hand size or finger length could impact the spatial distribution of key landmarks during gesture execution.

3.3. Exploratory data analysis

In this section, we perform an exploratory analysis of the data obtained. The exploratory analysis consists of checking the completeness and reliability of the data collected, defining the source of the data, and checking the distribution of the data. We present data such as the age of the users involved in constructing the dataset. Gender and occupation are also presented. The distribution of the raw and normalized data is also presented.

3.3.1. Demographic data analysis

In conducting demographic analysis on the dataset, the reported age of participants serves as a focal point. Figure 3.7 illustrates a normal distribution of participant ages, indicative of a typical spread within the dataset. In addition, it is observed that there are six people over forty years of age. However, it's essential to note that the accuracy of the models remains unaffected by these users as long as the gestures are executed proficiently. Thus, while deviations from the norm are observed, their impact on model accuracy remains negligible when gestures are performed effectively.

Having a normal distribution in the ages within the dataset presents an advantage. Because a normal distribution suggests that the majority of participants fall within a central range, this balance helps ensure that the dataset represents a diverse yet typical sample of the population under study. Additionally, a normal distribution facilitates statistical analysis and interpretation. Many statistical techniques and machine learning algorithms assume

normality in the data, allowing for more robust and reliable results. Moreover, a normal distribution simplifies the identification of trends, patterns, and relationships within the dataset, enabling researchers to draw meaningful insights and make informed decisions.

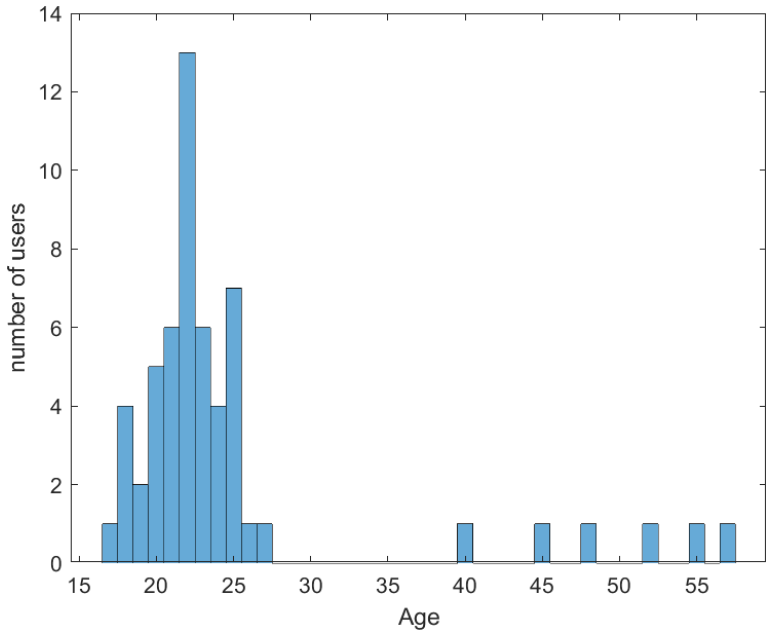

Figure 3.7: Representation of the age of users in the dataset

In the demographic analysis of the dataset, Figure 3.8 depicts the gender distribution among the 56 users included. The data reveals a relatively balanced representation of gender, with 27 male participants and 29 female participants. This gender parity is advantageous as it ensures diversity and inclusivity within the dataset, allowing for a comprehensive understanding of hand gesture recognition across different demographic groups. Furthermore, a balanced gender distribution facilitates gender-specific analyses and insights, enabling researchers to explore potential differences or similarities in gesture performance between males and females. Overall, the gender distribution depicted contributes to the dataset's robustness and validity, enhancing the reliability and applicability of findings derived from the analysis.

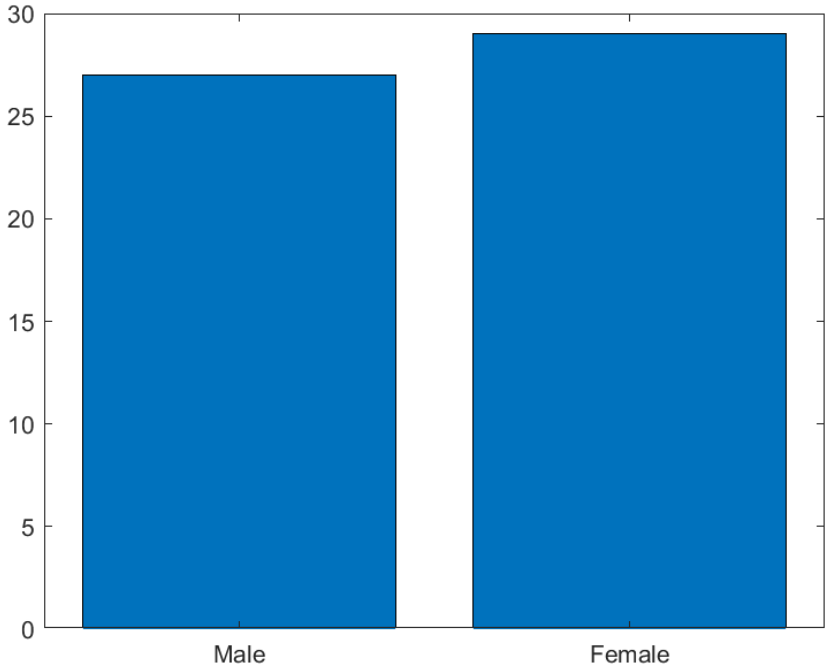

Figure 3.8: Gender of the users that are part of the dataset

The dataset presented in the Figure 3.9, comprising 56 users, it is revealed that 49 individuals do not report any injuries in their right limbs. Conversely, a smaller subset of users, totaling 7 individuals, have experienced some form of injury. This distribution highlights the prevalence of uninjured participants within the dataset, indicating a predominantly healthy sample population. However, the inclusion of users with injuries provides valuable insights into the potential impact of such conditions on hand gesture recognition performance. By considering both injured and uninjured users, the dataset captures a more comprehensive range of experiences and capabilities, enriching the analysis and facilitating a nuanced understanding of gesture recognition across diverse user profiles.

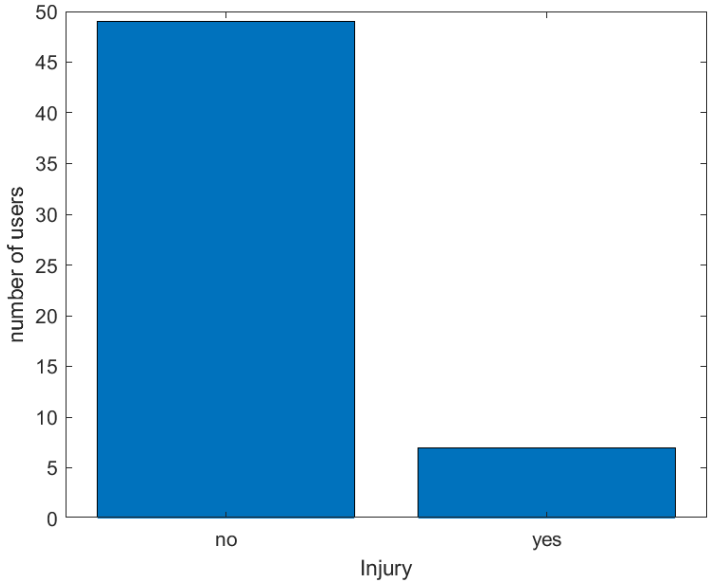

Figure 3.9: Injury users

For the analysis of the dataset **X**, a visualization of the data distribution based on the classes y is presented. Given that the dataset is multidimensional $\mathbf{X}_i = [x_1, x_2, ..., x_n]$, a t-distributed Stochastic Neighbor Embedding (tSNE) technique is employed to reduce the dimensionality to two components $\mathbf{X}_i = [c_1, c_2]$. . Figure 3.10 illustrates the distribution of raw data points corresponding to the five static gestures. This visualization allows for a clearer understanding of the inherent structure and clustering tendencies within the dataset, facilitating subsequent analysis and interpretation.

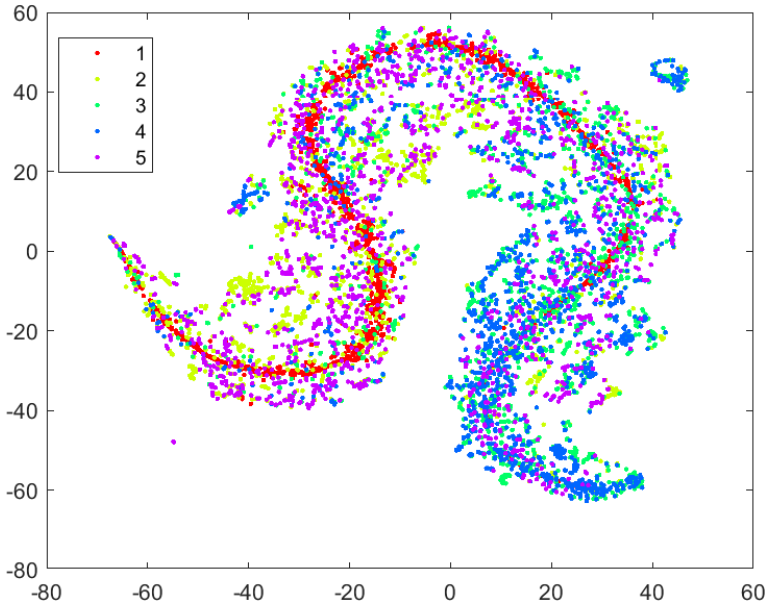

Figure 3.10: Distribution of the dataset with raw data

Visualizing data dispersion is integral to exploratory data analysis as it provides valuable insights into the underlying structure and patterns within a dataset. By representing data in graphical formats, researchers can discern trends, clusters, and outliers, facilitating the identification of biases, anomalies, and areas of interest. These visualizations aid in assessing feature importance, class imbalance, and separability issues, informing the selection of appropriate modeling techniques and preprocessing steps. Moreover, they help validate assumptions, refine data processing pipelines, and improve model performance. Overall, visualizing data dispersion enhances understanding, interpretation, and communication of dataset characteristics, contributing to more robust and reliable machine learning outcomes.

Once the dataset dispersion is visualized, data preprocessing steps can be performed to enhance the quality of the data representation. This includes techniques such as normalization, which aims to standardize the scale of features within the dataset. Normalizing the data helps to improve data dispersion and ensures that classes are better grouped, facilitating clearer separation and classification. As illustrated in Figure 3.11, the application of normalization techniques can lead to a more optimized dataset representation, enabling more effective analysis and modeling processes.

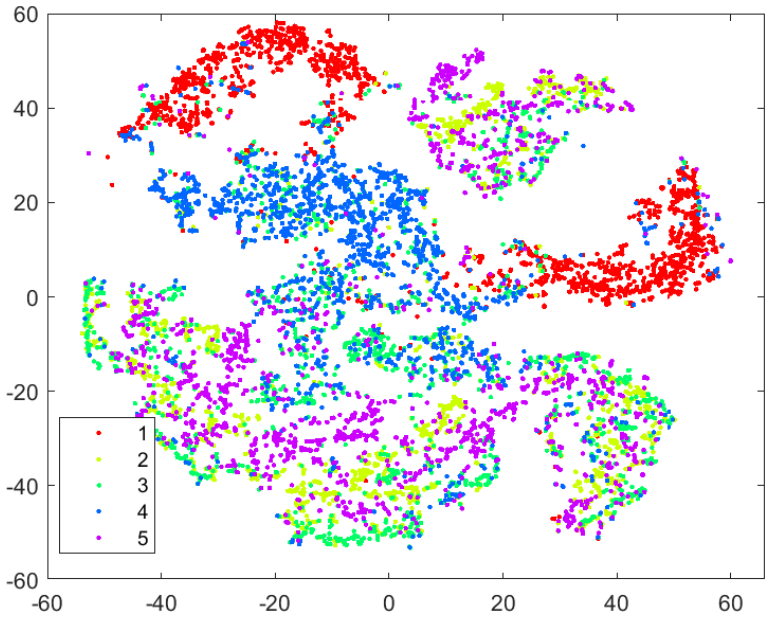

Figure 3.11: Distribution of the dataset with normalized data

Upon visualizing the dataset using the two-component representation, a histogram analysis reveals a distribution that tends toward a normal distribution, as shown in figure 3.12. This observation suggests that the data points are distributed symmetrically around the mean, with the majority clustered near the center and fewer instances towards the tails of

the distribution. The advantage of having a dataset that follows a normal distribution lies in its statistical properties, which facilitate various analytical and modeling processes. Normal distribution allows for the application of parametric statistical tests and inference methods, which assume normality in the data. Additionally, machine learning models perform optimally when the data conforms to a normal distribution. Therefore, having a dataset that exhibits a tendency towards a normal distribution enhances the reliability and interpretability of analytical results.

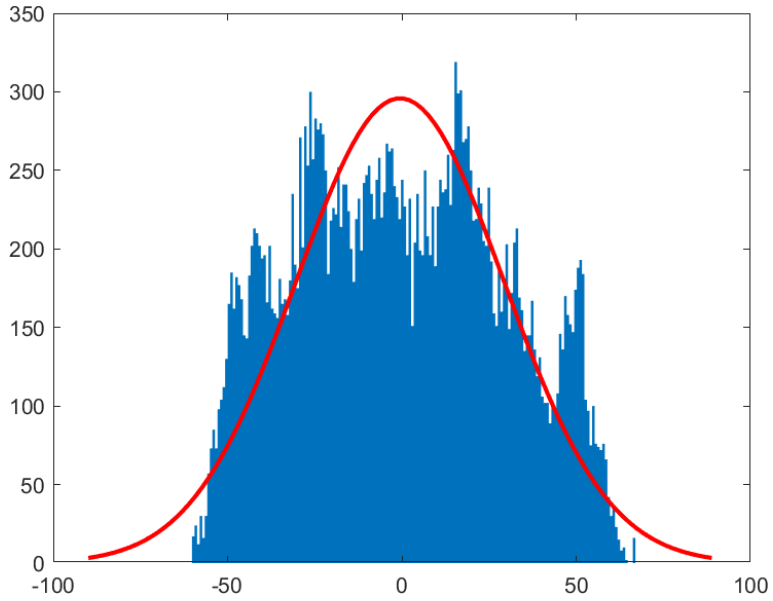

Figure 3.12: Normal distribution of the dataset

Similarly, Table 3.1 presents a correlation between the two dimensions and the probability value of the correlation coefficient being non-zero.

	Correlation	**p-value**
Value	0.107073	0.0000

Table 3.1: Correlation data and p-value from a two-dimensional dataset

3.4. Conclusions

This chapter shows how to build a dataset. For dataset construction, we need to define the type of sensor used. In this sense, we have described the difference, advantages and disadvantages between invasive and non-invasive sensors. After describing the types of sensors, we decided to use the Leap Motion Controller because it is an inexpensive, accurate, and

dedicated device for capturing hand movements.

We also developed a data acquisition interface. This interface is designed in MATLAB and acquires demographic data, spatial positions, and images. To avoid problems in the construction of the data set, we developed a data acquisition protocol. This protocol describes how the gesture should be executed by the user. Finally, the dataset was built with 56 volunteers; each volunteer performed five static and four dynamic gestures. In this sense, the dataset consists of spatial positions, directions, velocities, and images of the performed gesture. The dataset contains 15120 samples, 8400 static samples and 6720 dynamic samples.

Chapter 4

Evaluation of hand gesture recognition models using feature extraction and feature selection methods

Contents

4.1. Abstract

Hand gesture recognition is closely related to pattern recognition, where overfitting can occur when there are many predictors relative to the size of the training set. Therefore, we can reduce the dimensionality of the feature vectors through manual feature selection and

extraction techniques. This thesis proposes a study of feature selection and extraction methods for tested with traditional machine learning algorithms. The feature selection methods analyzed are: Maximum Relevance and Minimum Redundancy (MRMR), Sequential, Neighbor Component Analysis without Parameters (NCAsp), Neighbor Component Analysis with Parameters (NCAp), Relief-F, and Decision Tree (DT). Feature selection methods were fed with seventeen feature extraction functions, which return a score proportional to their importance. Finally, the feature extraction are ranked according to their scores and tested in classic machine learning algorithms such as ANN, SVM, kNN and DT.

4.2. Introduction

HGR is not a trivial problem because it is viewed as a pattern recognition problem [98], [99]. Also, the HGR problem is challenging to solve using mathematical or statistical models. Because to use mathematical models is necessary to know the complete problem comportment [100]. In contrast, statistical models need to know all variables and their comportment. For this reason, it is feasible to try to solve the HGR problem using deep learning methods or machine learning algorithms.

In this context, deep learning methods can address the HGR problem because these methods extract features automatically. However, deep learning methods have become complex to use because HGR systems need portability. In addition, deep learning methods require a large computational load and programming of complex models. The problem of the computational load has been solved by using GPUs. And the portability problem is solved using portable GPUs. However, portable GPUs require high power consumption, which is against portability.

In this sense, we need to return to traditional machine learning algorithms such as Artificial Neural Network (ANN), Support Vector Machine (SVM), k-Nearest Neighbors (KNN), Decision Trees (DT), among others. Machine learning algorithms typically operate in multidimensional environments, where data is usually represented as high-dimensional feature vectors. These feature vectors capture various data characteristics and represent the data in a format that the machine learning algorithm can analyze. By working in a multidimensional environment, machine learning algorithms can capture complex relationships and patterns within the data that can be used for tasks such as classification, regression, clustering, and dimensionality reduction. The machine learning models use feature selection and feature extraction as dimensionality reduction techniques.

Feature extraction is a fundamental process in machine learning that involves transforming raw data into a set of representative features or attributes. These features are selected or engineered to capture relevant information from the input data while reducing its dimensionality. This process often involves techniques such as mathematical transformations, dimensionality reduction methods, or domain-specific knowledge to extract informative features that can effectively represent the characteristics of the data. Feature extraction plays a crucial role in machine learning because it enables algorithms to work with more manageable and meaningful representations of complex data, leading to improved model performance and interpretability.

Feature selection is a process in machine learning aimed at identifying and choosing a subset of the most relevant features from the original set of input variables. feature selection involves directly choosing a subset of existing features without altering their values. The objective of feature selection is to improve model performance by reducing the dimensionality of the dataset and removing irrelevant, redundant, or noisy features that may hinder the learning process or lead to overfitting.

According to [101], The performance of machine learning models depends on the number of input variables utilized. While increased input variables can enhance accuracy, this relationship is limited by the finite nature of the dataset. As the number of features rises, model performance may improve but only up to a certain point. Beyond this threshold, additional features may lead to issues such as overfitting, where the model learns noise instead of true patterns. Consequently, model performance may decline due to unnecessary complexity. According to [102] the excess of variables can reduce the model performance because there may be a high correlation between variables. The definition of the variables with the highest descriptive load for the problem is not a trivial task. This is the reason why a lot of feature extraction and selection methods have been in development. Feature selection methods use some selection criteria, such as redundancy or data relevance, among others.

Feature selection methods are grouped into *filter*, *wrapper*, and *embedded* categories. **Filter** methods assign a score based on correlation, mutual information, or the classifier's performance on a single variable. **Wrapper** methods utilize the relevance of a feature subset to predict a machine learning model's performance accurately. Finally, **embedded** methods incorporate the selection of features into the training phase of algorithms such as DT [103].

In [104]–[106], the authors adopt the features selection as the variables retrieved by the leap motion controller. These features are the position and the orientation of the fingers and the hands. In addition, the authors use functions such as the mean, the standard deviation,

the correlation, the Shannon entropy, the kurtosis and the skewness as feature extraction. On the other hand, [90] presents a feature extraction of fingertip angles, fingertip distance, fingertip height, and fingertip position, and over these features use f-value, sequential feature selection, and random forest feature for its study. On the basis of this study, [107] mentions that the reduction of the number of features to a reasonable number is necessary, then they sort the more significant features with the f-score algorithm. The f-score calculates the ratio of the variance between classes and the variance within classes. Also, [108] uses the Gaussian Mixture Model to select features.

Within the framework of the filtering methods for feature selection, we analyze the method of the *Maximum Relevance Minimum Redundancy* (MRMR) and the *Sequentials*. For wrapper methods, we analyzed *Neighbors Component Analysis with lambda parameters* (NCAp) and *Neighbors Component Analysis without parameters* (NCAsp). Finally, for embedded methods, we analyze the *Relief-F* and *Decision Tree* (DT) methods for hand gesture classification and recognition. These methods are applied to a data set consisting of M observations and a set of 17 features. The features are obtained by applying 17 feature extraction functions to a raw data set retrieved from the Leap motion controller.

Feature selection methods are selected based on widespread adoption within the field. These methods are favored for their effectiveness in identifying and prioritizing relevant features essential for accurate classification and recognition tasks. Additionally, feature extraction functions are employed due to their incorporation of measures of central tendency and dispersion, enabling a more comprehensive characterization of the dataset. By leveraging these functions, the aim is to enhance the data categorization process and improve the discriminatory power of the resulting machine learning models.

4.3. Model description

In this chapter, we propose a generic machine learning model based on the spatial positions and directions of the hand to evaluate manual feature extraction and selection methods. The evaluation of these methods is performed in the classification module of the model. The classification module works with *parametric* and *non-parametric* classifiers.

A **parametric** classification algorithm models the function $P(y = 1|x; \beta)$ where β is the set parameters. The training of parametric algorithms consists of adjusting the values of β using an optimization algorithm [109]. This work uses *ANN* and *SVM* as parametric algorithms.

ANN is inspired by biological neural networks. The biological neuron consists of dendrites, soma, and axon. By analogy, the dendrites of a biological neuron correspond to the connections made between the input variables of an artificial neuron and the node where the mathematical operations are executed. This node corresponds to the soma and allows the neuron to be activated and generate a response. The response would correspond to the output function of the artificial neuron, equivalent to the axon of the biological neuron. Figure 4.1 illustrates the biological neuron and the artificial neuron.

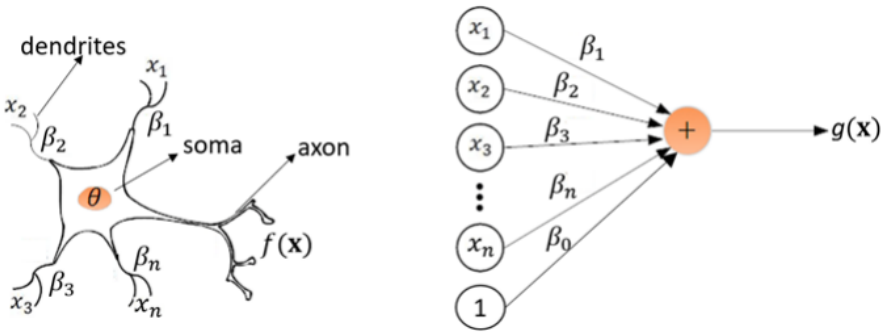

Figure 4.1: Analogy of a biological neuron and an artificial neuron

An artificial neuron is called a perceptron. The perceptron is a linear classifier that results from calculating the scalar product of the input vector **x** and the synaptic weights β, according to the following equation $g(\mathbf{x}) = f(\beta_0 + \beta_1 x_1 + ... + \beta_n x_n)$, where f represents the activation function [110].

ANN is a result of connecting several perceptrons grouped in layers, as shown in Figure 4.2. ANN is a non-deterministic algorithm due to the random weight initialization. ANNs are widely used because of their high parallelism, high approximation capabilities, high noise tolerance, and their good capabilities of learning and generalization [111].

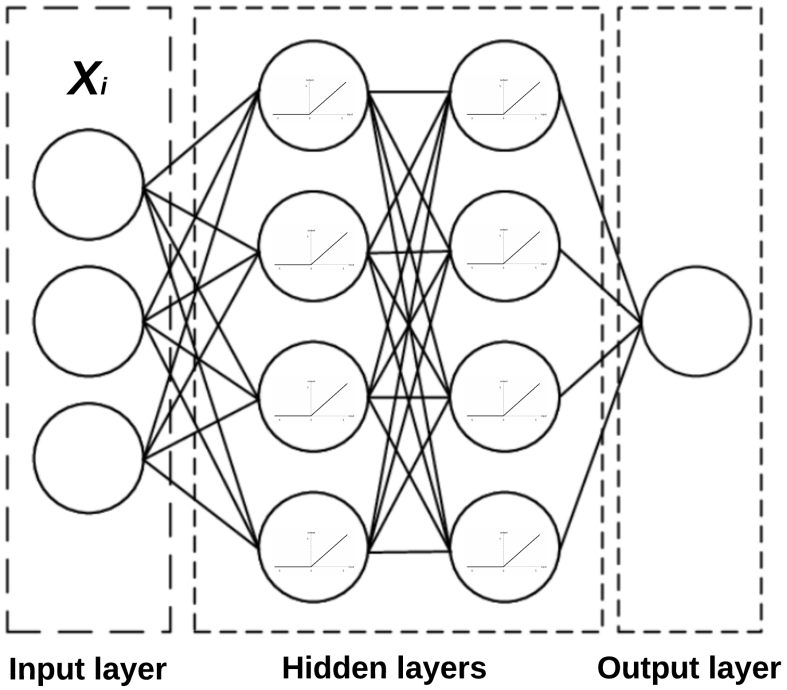

Figure 4.2: Artificial neural network

SVM iSVM is a supervised learning algorithm utilized for classification and regression tasks in machine learning. It works by constructing a hyperplane or a set of hyperplanes in a high-dimensional space to separate instances of different classes. The primary objective of SVM is to find the hyperplane that maximizes the margin between the classes, thereby improving the model's generalization performance. SVM is particularly effective in handling high-dimensional data and is known for its versatility in dealing with both linearly separable and non-linearly separable datasets through the use of kernel functions. Furthermore, SVM is initially designed as a binary classifier, distinguishing between two classes. To extend it to multiclass classification tasks, various strategies like one-vs-one or one-vs-all are employed, enabling SVM to handle multiple classes effectively. [112]. SVM uses linear functions such as $\mathbf{wx}_i + b \geq 1$, where **w** and **x** are vectors. The closest vectors to the hyperplane are called support vectors.

Non-parametric classifier is a type of machine learning algorithm that doesn't make explicit assumptions about the underlying distribution of the data. Non-parametric classifiers learn directly from the training data, adjusting their complexity based on the data's characteristics. These classifiers are particularly useful when the underlying data distribution is complex or unknown, as they can adapt more flexibly to different types of data without relying on predefined parameters or assumptions. [113]. This thesis uses *KNN* and *DT* as

non-parametric algorithms.

KNN is called a lazy algorithm. It is a non-parametric method because it does not involve parameter adjustment or estimation. Also, it is a deterministic algorithm that identifies dynamically k observations of a dataset that are similar to a new observation. To identify similarities kNN uses a distance metric. The distances metrics could be the Euclidean distance, Minkowski, Mahalanobis or the dynamic time warping (DTW) method [114], [115]. KNN defines the number of neighbors according to the value of K. KNN represents a higher similarity when the distance between samples is small, so they are highly likely to belong to the same label [116].

DT is a machine learning algorithm used for classification and regression. This algorithm selects an attribute from a set of training instances. The attribute selected is a tree node. Next, the decision tree is built using a training instance and the selected attribute. Each internal node tests an attribute x_i, each branch assigns an attribute value, and each leaf gives a class. The DT is very susceptible to changes in dataset values. The tree is traversed from root to leaf to classify a new input [117].

These algorithms rely on various data inputs to make predictions or classifications, making the quality and relevance of the input signals a crucial factor in their performance. When applied to hand gesture recognition tasks, these algorithms require precise and meaningful signals to learn and classify different gestures effectively. The signals retrieved from the LMC are vital inputs for training and testing these machine-learning models. Figure 4.3 shows a scheme of the HGR model using infrared information retrieved by LMC. The data are filtered and normalized in the preprocessing module. The feature extraction module uses methods that will be discussed later in this chapter, while the classification module use algorithms such as ANN, SVM, KNN, and DT.

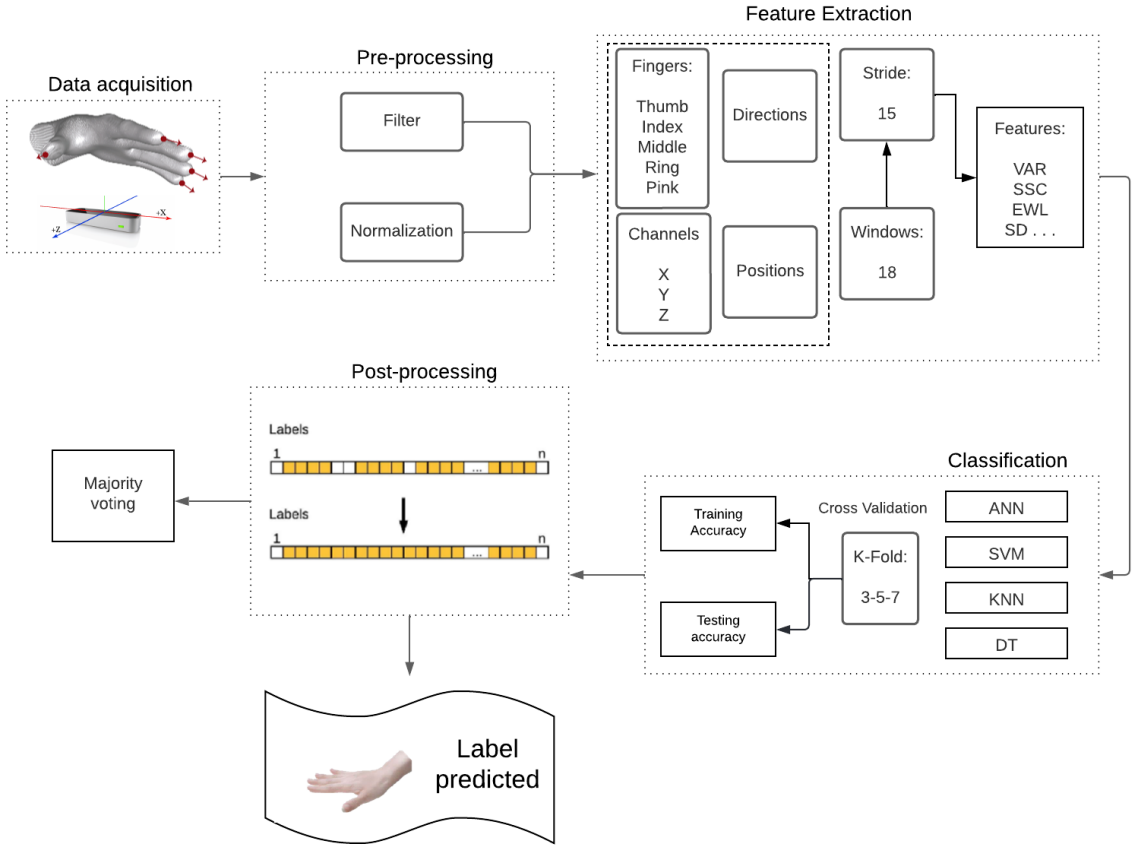

Figure 4.3: General scheme of HGR model using infrared information

For this thesis, we used two types of signals for the **data acquisition** module using the LMC. The first signal is points of spatial positions, and the second is directions. The **first signal** corresponding to the points of the spatial positions is represented as time series. The spatial position of the fingers at time *t* is represent using:
$\mathbf{P}_t = [p^{(x)}_{(1,t)}, p^{(y)}_{(1,t)}, p^{(z)}_{(1,t)}, ..., p^{(x)}_{(5,t)}, p^{(y)}_{(5,t)}, p^{(z)}_{(5,t)}]^{(leap)}_t$, being $[p^{(x)}_{(i,t)}, p^{(y)}_{(i,t)}, p^{(z)}_{(i,t)}]$ the vector with the spatial positions of the *i-th* finger concerning the sensor coordinate axes. The **second signal** corresponding to the directions represented using:
$\mathbf{D}_t = [d^{(x)}_{(1,t)}, d^{(y)}_{(1,t)}, d^{(z)}_{(1,t)}, ..., d^{(x)}_{(5,t)}, d^{(y)}_{(5,t)}, d^{(z)}_{(5,t)}]^{(leap)}_t$, being $[d^{(x)}_{(i,t)}, d^{(y)}_{(i,t)}, d^{(z)}_{(i,t)}]$ the vector with the directions of the *i-th* finger concerning the sensor coordinate axes.

The **preprocessing** module consists only of normalized spatial positions. For the **feature extraction** module, we use the window division technique. We define a window size of $w = 18$ and a step size of $s = 15$.

Each observation $\mathbf{p}_i = [x_1, ..., x_n]$, of the matrix **P** is divided into windows $\mathbf{O}_w = [\mathbf{x}_i, \mathbf{x}_{i+1}, ..., \mathbf{x}_j]$, where $|w| = j - i + 1$. The feature extractor functions $f_{extractor}$ are applied to the data of

each window $\mathbf{O}_w$; obtaining $\mathbf{fe} = f_{extractor}(\mathbf{O}_w)$.

The union of the features of each channel forms the feature vector of the window $\mathbf{fe}_w = [\mathbf{fe}_1, ..., \mathbf{fe}_k]$. Figure 4.4 shows the feature extraction of 3 channels of the first window.

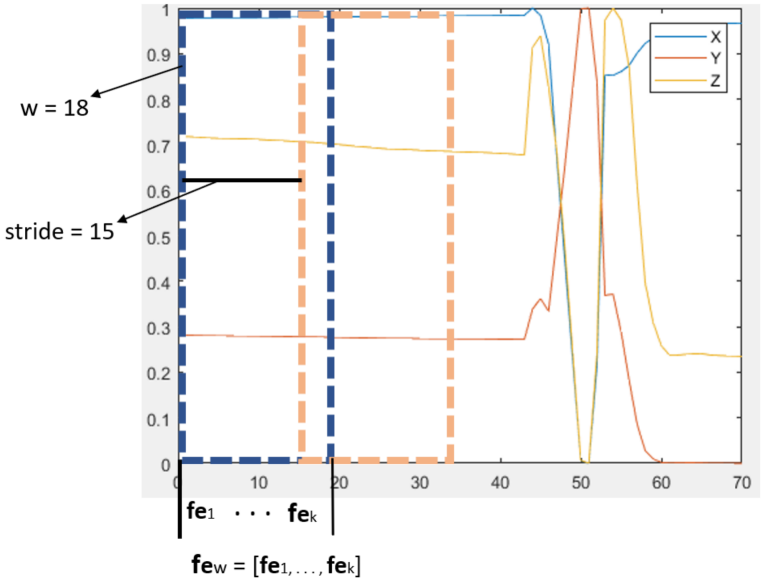

Figure 4.4: Feature extraction of three channels from window 1

In the **classification** module, we use a feedforward **ANN** with two hidden layers. The first hidden layer uses 25 neurons and also uses ReLU as an activation function. The second hidden layer uses 15 neurons and logsig as the activation function. The input of ANN is the number of features according to the number of combinations of feature selection functions. Since the optimization functions use *cross-entropy*, *gradient descent* is used for of weight adjustment. The **cross-entropy** is a loss function used to measure the performance of classification models, particularly in probabilistic contexts. It calculates the difference between the true label and the predicted probability distribution, penalizing confident but incorrect predictions more heavily. By minimizing cross-entropy, the model improves its ability to accurately predict class probabilities. The **gradient descent** method is an optimization algorithm used to minimize the loss function in machine learning models by iteratively adjusting the model parameters. It calculates the gradient of the loss function with respect to the parameters and updates the parameters in the opposite direction of the gradient to find the optimal values. This process continues until the loss function converges to a minimum value, indicating the best-fit model parameters. In addition, ANN uses 2000 epochs and

a regularization factor for weight decay of 1.0e1. This architecture was selected based on empirical testing, which compared it with other architectures and found it to have the lowest possible model error. The selection process employed an accuracy estimation paradigm to ensure the most accurate model was chosen.

The **SVM** classifier uses a Gaussian kernel, with a scale of order 10. the parameter scale refers to a scaling factor applied to features before fitting the model. It helps normalize the input data to ensure that all features have a similar scale, which can improve the convergence of the optimization algorithm and the overall performance of the SVM model. Typically, scaling is essential when the features have different units or scales, preventing certain features from dominating others during training. **KNN** was set with k equal to 3 with the Euclidean distance. **DT** uses the technique of information gain to build the tree. Also, this algorithm uses 100 levels of depth for training.

The **postprocessing** module fine-tunes the label vector resulting from the evaluation of each window in the classification module. Given that the elements of the label vector may differ, each element $\hat{\mathbf{l}} = (\hat{l}_1, \hat{l}_2, \cdot, \hat{l}_n)$ is scrutinized individually. If the label at position $\hat{l}_t$ differs from its immediate predecessor $\hat{l}_{t-1}$, but matches the label immediately before and after it $\hat{l}_{t-1} == \hat{l}_{t+1}$, the evaluated label is modified to match the previous label $\hat{l}_t := \hat{l}_{t-1}$.

4.4. Feature extraction methods

For the present work, we defined 17 feature extraction functions. These functions are based on statistical measures of central tendency, dispersion, amplitude, wavelength, among others. The feature extraction functions used are described below.

- Variance (VAR) measures the signal amplitude and the power
- Root mean square (RMS) is a meaningful way of calculating the average of values over a period of time
- Mean absolute value (MAV) define the average of the summation of absolute value of signal
- Enhanced mean absolute value (EMAV) is an extension of MAV define a p value this value is used to select a region of the signal
- Modified mean absolute value (MMAV) is an extension of MAV this assign the weight window function

- Modified mean absolute value 2 (MMAV2) is another extension of MAV feature by assigning the continuous weight window function
- Difference absolute standard deviation value (DASDV) is the square root of the average of the difference between the squared contiguous values
- Enhanced wavelength (EWL) is an extension of WL define a p value this value is used to select a region of the signal
- Average amplitude change (AAC) measures the average change of the signal amplitude
- Wavelength (WL) can be calculated by simplifying the cumulative length of waveform summation
- Slope sign change (SSC) determines the number of times in which the number of wave form changes sign
- Detector log (DL) is good at estimating the exerted force
- Pulse percentage rate (MYOP) this function is adapted by leap motion controller signal
- Amplitude Willinson (WAMP) acts as an indicator of the firing of motor unit potentials
- Simple square integral (SSI) is defined as the summation of square values of signal amplitude
- Standard deviation (SD)
- mean value (MV) [118].

4.5. Feature selection methods

To assess the effectiveness of the feature selection methods, a fresh dataset is derived from the one detailed in Chapter 3. This new dataset captures hand movements by summing each finger's components' X, Y, and Z spatial positions. The structure of the dataset is MxN, where M is the total number of observations and N is the number of features describing the movement of the hand. Each feature extraction function described in the preview section is applied to the dataset, and a matrix Mx17 is obtained. Each data obtained by the feature extraction function represents a predictor.

4.5.1. Filter methods

The **minimal-redundancy maximum-relevance** algorithm belongs to the family of wrapper methods for feature selection. The algorithm finds a set of mutually and maximally exclusive features. The algorithm quantifies redundancy and relevance, as from the dependence of a pair of features and the dependence of a variable against the response variable [119]. The algorithm for maximizing relevance examines the relationship between a variable and the response variable. In [120] present the equation $\mathbf{V}_S = \frac{1}{|S|} \sum_{x \in S} I(x,y)$, where $\mathbf{V}_S$ represents the maximization value, $|S|$ represents the set of features or predictor variables, (x,y) represents the predictor variable and the response variable, respectively, and *I* represents the indicator function of the dependence between the predictor and the response variable.

While for the minimization, the dependence between the pair of predictor variables is observed, as presented in the following equation $\mathbf{W}_S = \frac{1}{|S|^2} \sum_{(x,z) \in S} I(x,z)$. Where $\mathbf{W}_S$ represents the value of minimization of the dependence between the predictor variables, and (x,z) represents the predictor variables. The *I* indicator function measures the correlation between predictor variables. $\mathbf{V}_S$ measures the maximum relevance of a variable against the response variable. It analyzes the extent to which a variable is dependent on the target outcome, providing insights into its predictive power. $\mathbf{W}_S$ quantifies the redundancy among predictor variables, and aims to minimize redundancy, ensuring that the selected variables provide unique and complementary information for predictive modeling.

The algorithm of MRMR returns an index and an associated score. The score defines the importance of the predictor variable. In this context, the MRMR algorithm is fed with data extracted from the functions in the following order: MAV, EMAV, MMAV, MMAV, MMAV2, VAR, RMS, DASDV, SD, MV, ACC, WL, EWL, LD, SSC, MYOP, WA, SSI. Once the MRMR algorithm processes the dataset with the feature extraction functions, the algorithm returns the score of the most significant predictor variables, as shown in figure 4.5. The order of the variables corresponds to VAR, SSC, EWL, SD, WA, WL, LD, DASDV, EMAV, MYOP, MAV, ACC, MMAV, SSI, MMAV2, MV, RMS.

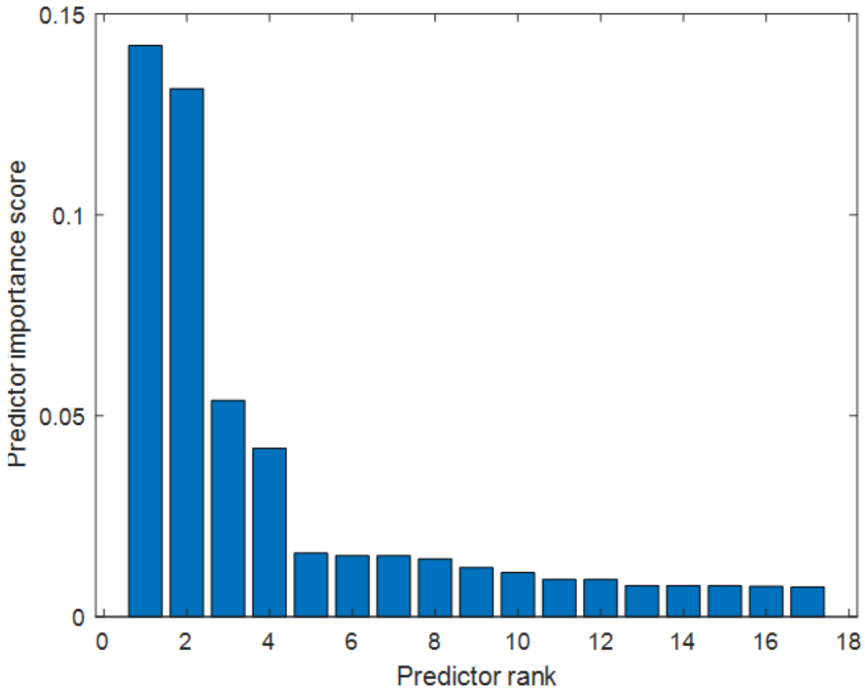

Figure 4.5: Order of feature extraction functions after running feature selection functions with the algorithm MRMR. The order of the functions on the X-axis are as follows VAR, SSC, EWL, SD, WA, WL, LD, DASDV, EMAV, MYOP, MAV, ACC, MMAV, SSI, MMAV2, MV, RMS.

Sequential selects a subset of features from the data matrix **X** that best predicts the data in *y*, where **X** is a matrix of data and *y* are the classes. This sequential feature selection method iteratively chooses features until there is no further enhancement in prediction accuracy. It relies on a predefined criterion function to identify the optimal subset of features. The process involves two main steps: initially splitting the dataset into training and testing sets, followed by a cross-validation procedure. During evaluation, the method calculates the mean score based on the summed values returned by the criterion function, divided by the total number of test observations. This mean score serves as the basis for ranking candidate feature subsets. Moreover, the method evaluates classification performance by considering the number of misclassified observations.

After computing the mean criterion values for each candidate feature subset, the method chooses the candidate feature subset that minimizes the mean criterion value. This process continues until adding more features does not decrease the criterion. In this thesis, the method is fed with the data matrix Mx17, the values of the features correspond to the values of the randomly ordered feature selection functions. Table 4.1 presents in the first column the

Initial order of feature extractor functions	Order according to the score after applying the sequential feature selection method	Score
MAV	MYOP	0.000768282
EMAV	LD	0.000723214
MMAV	EWL	0.000658022
MMAV2	WL	0.00061593
VAR	MAV	0.000597222
RMS	SSC	0.000588435
DASDV	MMAV	0.000586876
SD	EMAV	0.000584325
MV	MV	0.000583475
SSC	VAR	0.000583333
WL	MMAV2	0
EWL	RMS	0
LD	DASDV	0
AAC	SD	0
MYOP	AAC	0
WA	WA	0
SSI	SSI	0

Table 4.1: Score of feature extraction functions using sequential method

initial functions, while the second column presents the ordered feature selection functions based on the score returned by the method.

4.5.2. Wrapper methods

The **neighbor component analysis** (NCA) aims to optimize a linear transformation of the feature space to maximize the accuracy of the models. The process of NCA is similar to KNN, where k equals one. NCA also has a distance metric, and for measuring the distance, NCA selects a $\mathbf{x}_I$ randomly as a reference point to other feature vectors $\mathbf{x}_j$. It obtains $\mathbf{d}_{i,j} = (\mathbf{x}_i, \mathbf{x}_j)$, where **d** represents the distance. NCA's effectiveness lies in its direct optimization of the k-NN classifier's performance and interpretable insights into feature importance. NCA iteratively updates the transformation matrix based on observed classifier performance. It measures neighborhood relationships between data points to evaluate the impact of different feature subsets on classification accuracy. NCA adjusts the transforma-

tion matrix parameters to increase the likelihood of correct class predictions for each data point based on its neighbors. The iterative process continues until convergence, ensuring no further improvement in classification accuracy.

In this thesis we experimented with two versions of NCA. One version with parameter setting called *NCAp* and another version without parameter setting called *NCAsp*. Neighbor Component Analysis with Parameters **(NCAp)** utilizes various parameters to adjust the classification accuracy, including the regularization value (lambda), optimization function, fit method, and standardization process [121]. To determine the optimal lambda value, we create an array of 50 evenly spaced values ranging from 0, incremented by 3. Each value is scaled by the standard deviation of the total observations and divided by the total number of observations. Subsequently, the NCA algorithm is executed for each value in the array, employing a k-fold cross-validation with a value of 5, to mitigate bias. The optimization function employed is stochastic gradient descent. Finally, we present the order of the most significant feature extractor functions reported according to Table 4.2.

Functions	**Score**
LD	3.819
EMAV	2.869
MMAV2	2.820
MMAV	2.413
SD	2.256
DASDV	2.229
MYOP	1.865
AAC	1.760
WL	1.665
EWL	1.665
MAV	1.489
MV	1.489
VAR	0.045
SSI	0.029
RMS	0.029
SSC	6.06exp-15
WA	2.24exp-58

Table 4.2: Score of feature extraction functions using NCAp with parameters.

Table 4.2 reveals a clear trend in model performance based on the sequential application of feature extraction functions selected by the NCAp method. Initially, there is a discernible improvement in performance with the incorporation of the first 6 functions. Subsequently, performance appears to plateau as additional functions are included from the 7 to the 15 position. However, beyond this point, there is a noticeable decline in model performance.

A second model is also generated using Neighbor Component Analysis Without Parameters **(NCAsp)**. The input for this method is the observations and the classes as parameters. The observations are the matrix (dataSetFeatures) and their respective labels *y*. Table 4.3 presents a score of the most significant predictor functions.

Functions	**Score**
LD	9.202
MYOP	8.740
WL	3.577
EWL	3.381
SSI	2.246
EMAV	6.97exp-3
MMAV2	2.97exp-3
SD	2.65exp-3
MMAV	1.18exp-3
MAV	9.23exp-4
MV	9.23exp-4
DASDV	2.76exp-7
VAR	3.12exp-9
RMS	2.74exp-9
AAC	4.56exp-10
SSC	3.26exp-41
WA	2.83exp-159

Table 4.3: Score of feature extraction functions using NCAsp without parameters.

According to Table 4.3, we can observe a pattern where the performance gradually increases with the incorporation of the first five feature extraction functions. However, beyond this point, as additional feature extraction functions are included, there appears to be a decline in performance.

4.5.3. Embedded methods

Another feature selection algorithm used in this thesis is **Relief-F**. It operates by evaluating the importance of each feature based on its ability to distinguish between observations of different classes. The algorithm assigns a weight to each feature, representing its relevance in classification. Relief-F works by iteratively sampling observations from the dataset and comparing the feature values of each observation with those of its nearest neighbors from the same and different classes. By computing the differences in feature values, Relief-F assesses how well each feature discriminates between observations of the same and different classes. Features that consistently exhibit large differences between observations of different classes are assigned higher weights, indicating their importance in the classification process. In this way, Relief-F effectively selects the most discriminative features while disregarding irrelevant or redundant ones, ultimately improving the performance of machine learning models by reducing dimensionality and focusing on the most informative features [122].

In [120], they present the equation to update the weights when the class of the selected observation and the predicted observation class are the same. The equation is $\mathbf{w}_j^i = \mathbf{w}_j^{i-1} - \frac{\Delta_j(\mathbf{x}_r * \mathbf{x}_q)}{m} * d_{rq}$. Where $\mathbf{w}_j^i$ are the weight of feature j at observation i, $\mathbf{w}_j^{i-1}$ are the weight of feature j at observation $i-1$, $\Delta_j(\mathbf{x}_r * \mathbf{x}_q)$ is the difference between the jth feature value of the selected observation $\mathbf{x}_r$ and jth feature value of the predicted observation $\mathbf{x}_q$, m is the total number of features in the dataset, and d_{rq} is the distance between the selected observation $\mathbf{x}_r$ and the predicted observation $\mathbf{x}_q$.

Likewise, updating the weights is required when the feature vectors $\mathbf{x}_r$ and $\mathbf{x}_q$ belong to classes with different labels, as indicated by the following equation. $\mathbf{w}_j^i = \mathbf{w}_j^{i-1} + \frac{p_{yq}}{1-p_{yr}} * \frac{\Delta_j(\mathbf{x}_r * \mathbf{x}_q)}{m} * d_{rq}$. Where $\mathbf{w}_j^i$ represent the weights of the predictor for *ith* iteration $\mathbf{p}_{yr}$ represent a priory probability of the class which $\mathbf{x}_r$ belongs, p_{yq} represent a priory probability of the class which $\mathbf{x}_q$ belongs, *m* represents the iterations number given for updating the weights. Finally, $\Delta_j(\mathbf{x}_r * \mathbf{x}_q)$ is a difference in the predictors' values between the observations $\mathbf{x}_r$ and $\mathbf{x}_q$.

The algorithm is fed with the feature extractor functions without a specific order. After applying the algorithm, we obtain the following ranking. LD, SSC, EWL, MYOP, DASDV, MMAV, ACC, WL, EMAV, SD, MAV, MV, RMS, SSI, VAR, MMAV2, WA. Figure 4.6 presents the ranking of the feature extractor functions.

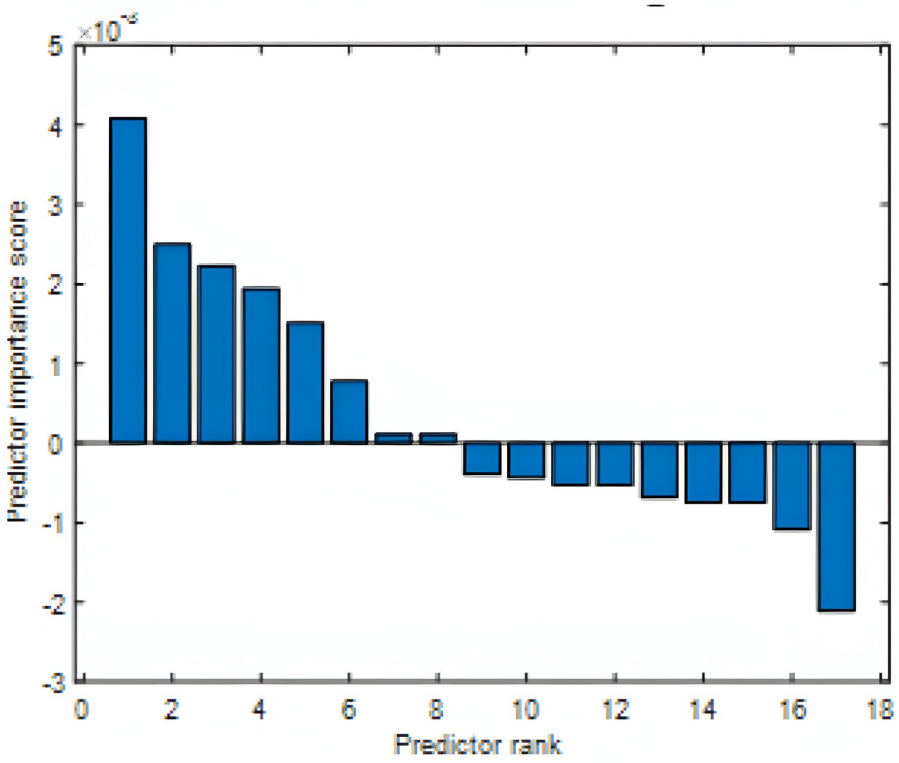

Figure 4.6: Ranking of most important feature using Relief-F. The order of the functions on the X-axis are as follows LD, SSC, EWL, MYOP, DASDV, MMAV, ACC, WL, EMAV, SD, MAV, MV, RMS, SSI, VAR, MMAV2, WA.

In the context of using decision trees (DT) as a feature selection method, a decision tree is represented by the function $DT(textbfx)$, where **x** represents the input feature vector. The decision tree recursively partitions the feature space into subsets based on the values of the features. At each node t, the algorithm selects the feature x_j that best splits the data into subsets $\mathbf{X}_t^L$ and $\mathbf{X}_t^R$ based on a splitting criterion such as entropy or Gini impurity, where *L* is left and *R* is right. The algorithm continues recursively until a stopping criterion is met, such as reaching a maximum depth or minimum number of samples per leaf. The resulting decision tree implicitly ranks features based on their importance in determining the target variable, with features closer to the root node being deemed more important. Table 4.4 shows the order of the feature extraction functions based on the selection of the most relevant attribute using DT

Initial order of feature extractor functions	**Order according to the score after applying the sequential feature selection method**
MAV	LD
EMAV	MYOP
MMAV	WL
MMAV2	EWL
VAR	SSI
RMS	EMAV
DASDV	MMAV2
SD	SD
MV	MMAV
AAC	MAV
WL	MV
EWL	DASDV
LD	VAR
SSC	RMS
MYOP	AAC
WA	SSC
SSI	WA

Table 4.4: Order of feature extraction functions using the decision tree.

4.6. Evaluation of feature extraction and feature selection methods

The evaluation consisted of ordering the feature extractor functions according to the score provided by each feature selection method proposed. Then, the dataset is divided into windows. The windows have a length of 20 with steps of 15. The function with the highest score reported is applied to each window, and at the end, a vector of labels is obtained. The label that the classifier returns is chosen by a majority vote. Then, the same process is executed, but with the feature selection function with the second highest score, and so on, until all feature selection functions are combined. This process is repeated 10 times, and finally, a 10x17 matrix is obtained, where each cell corresponds to the accuracy. The average accuracy across the 10 iterations is calculated, yielding a comprehensive performance overview. The maximum accuracy achieved across all experiments is then determined. The index of this maximum value corresponds to the optimal combination of feature selection functions, indicating the most effective approach for the given dataset and classification task, as presented in the following pseudocode of the table 4.5.

```
1   Fs = [VAR, SSC, EWL, SD, WA, WL, LD, DASDV, EMAV, MYOP, MAV,
    ACC, MMAV, SSI, MMAV2, MV, RMS];
2   Win = 20;
3   Step = 15;
4   Ds = data;
5   i = 1;
6   Alg = [ANN, SVM, KNN, DT];
7   for repetition = 1 :10
8       until (i = 1 : length(Fs))
9           for (j = 1 : (size(ds,2))/Win)
10              acc(j) = Alg(Fs(1:i)(ds(Win)));
10          end
11          acc(i,j) = mode(acc);
12      end
13  end
```

Table 4.5: Algorithm 1, returns the accuracy of the evaluation of feature extraction functions on ANN, SVM, kNN, DT classifiers.

In Algorithm 1, Fs denotes the array of feature extraction functions, arranged based on

the scores provided by the feature selection methods. Meanwhile, ds represents the dataset, and Alg signifies the array of classifiers utilized for evaluating the feature extraction functions.

Table 4.6 shows a summary of the experiments. It shows the number of combinations represented by the variable idx. It also shows the maximum accuracy and the standard deviation of each model trained, and tested. Also, in the recognition column the processing time is presented.

		Training			**Testing**			**Recognition**			
		idx	Acc	std	idx	Acc	std	idx	Acc	std	time
MRMR	ANN	17	98.332	0.180	13	92.759	1.454	9	82.271	1.820	82.411
	SVM	17	98.948	0.135	9	91.413	1.619	9	79.915	2.456	13.119
	KNN	8	91.600	1.090	8	91.600	1.089	9	77.091	1.901	0.865
	DT	16	89.903	0.949	16	89.903	0.949	16	75.288	2.295	0.303
NCAp	ANN	17	98.341	0.185	10	92.736	1.482	13	83.335	2.314	80.065
	SVM	17	98.948	0.135	10	91.527	1.429	12	80.121	3.004	12.236
	KNN	12	91.018	1.883	12	91.018	1.883	13	77.667	2.662	0.683
	DT	13	89.903	1.801	13	89.903	1.801	17	75.364	2.319	0.263
NCAsp	ANN	17	98.319	0.182	4	92.239	1.682	15	81.674	2.568	90.904
	SVM	17	98.948	0.135	9	91.321	1.237	12	79.864	3.051	15.756
	KNN	4	90.121	1.141	4	90.121	1.141	4	75.046	3.081	0.695
	DT	11	89.903	1.813	11	89.903	1.813	17	75.333	2.267	0.264
ReliefF	ANN	17	98.332	0.131	13	92.878	1.773	13	83.403	2.508	88.053
	SVM	17	98.948	0.135	10	91.158	1.742	12	80.379	2.540	12.526
	KNN	3	91.394	0.782	3	91.394	0.782	13	76.591	2.787	0.661
	DT	13	89.842	1.326	13	89.842	1.326	13	76.136	3.136	0.238
Sequentials	ANN	16	98.471	0.114	14	93.079	1.579	14	83.515	1.854	67.565
	SVM	17	98.948	0.177	15	90.776	1.617	15	79.818	3.102	12.129
	KNN	15	90.752	0.894	15	90.752	0.894	14	76.652	2.766	0.595
	DT	14	89.867	1.143	14	89.867	1.143	16	75.394	2.4061	0.191
DT	ANN	16	98.504	0.120	6	93.055	1.506	16	83.227	2.305	44.792
	SVM	17	98.948	0.135	7	91.382	1.646	15	79.864	3.298	6.853
	KNN	13	90.824	0.789	13	90.824	0.789	15	76.621	2.327	0.591
	DT	15	89.842	1.218	15	89.842	1.220	15	75.379	2.291	0.186

Table 4.6: Overview of maximum training, testing, and recognition accuracies achieved by feature selection methods in classification algorithms. Also included are standard deviations, number of feature combinations, and processing times.

Figure 4.7 shows the testing accuracy grouped by each method and evaluated for each

proposed classification algorithm. In addition, the number of combinations of feature extractor functions with which the methods reach their maximum value is presented.

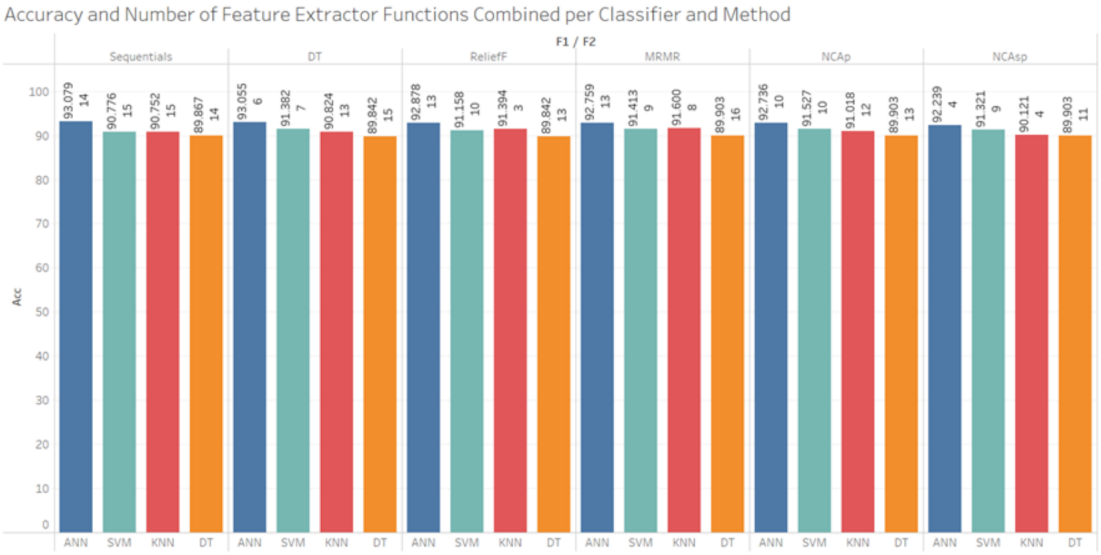

Figure 4.7: Summary of the maximum training, testing, and recognition accuracy of feature selection methods evaluated in classification algorithms. Standard deviation, number of feature combinations, and processing time.

In Figure 4.7, it is evident that the disparities in reported accuracies are minimal, with overlapping standard deviation values. This tells us that there is no significant difference between the methods in the evaluated algorithms. This leads us to perform a statistical hypotesis testing to determine significant differences and to define the method, the combination of feature selection functions, and the best-performing algorithm. Since there are 10 repetitions of the experiments, a test of variance hypothesis is performed to demonstrate whether there is a statistically significant difference in the evaluation of the feature selection methods. Figure 4.8 shows that all the methods evaluated with ANN have the maximum accuracy score. It is also evident that only with the ANN algorithm the data present a variation, this is due to the fact that the algorithm introduces a variability due to the stochasticity in the initialization of its weights. The other methods do not present a variability, so they could be compared directly. Now, with the ANN data, first, the normality test is performed using the Shapiro-Wilk normality test. Then, we check if we have a homogeneity of variances. Finally, an ANOVA is performed.

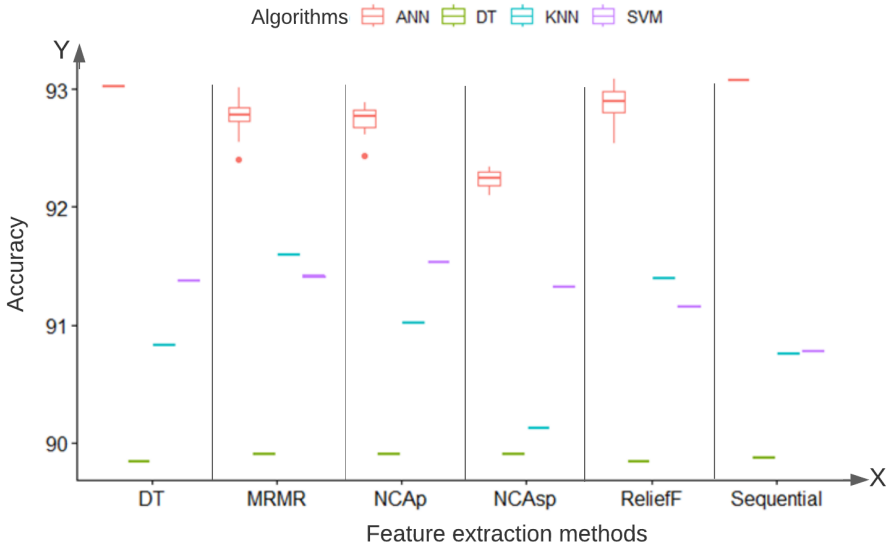

Figure 4.8: Variability of the data and the algorithms with the highest accuracy values

Figure 4.9 offers a detailed breakdown of the performance results obtained from each feature selection method, categorized by the respective algorithms they were evaluated with. Notably, the evaluation reveals that across all algorithms, the ANN algorithm consistently achieves the highest accuracy rates when paired with various feature selection methods. Within the ANN framework, it is evident that both the Sequential and Decision Tree (DT) methods stand out with remarkable accuracies, boasting 93.079% and 93.055%, respectively. However, a critical aspect to consider in this analysis lies in the number of feature extraction functions employed to attain these accuracy levels. Specifically, Sequential utilizes a larger set of 14 distinct functions, while DT achieves comparable results with a leaner selection of only 6 feature extraction functions. This nuanced comparison underscores the significance of selecting an optimal feature selection strategy tailored to the specific algorithmic context, balancing accuracy requirements with computational efficiency.

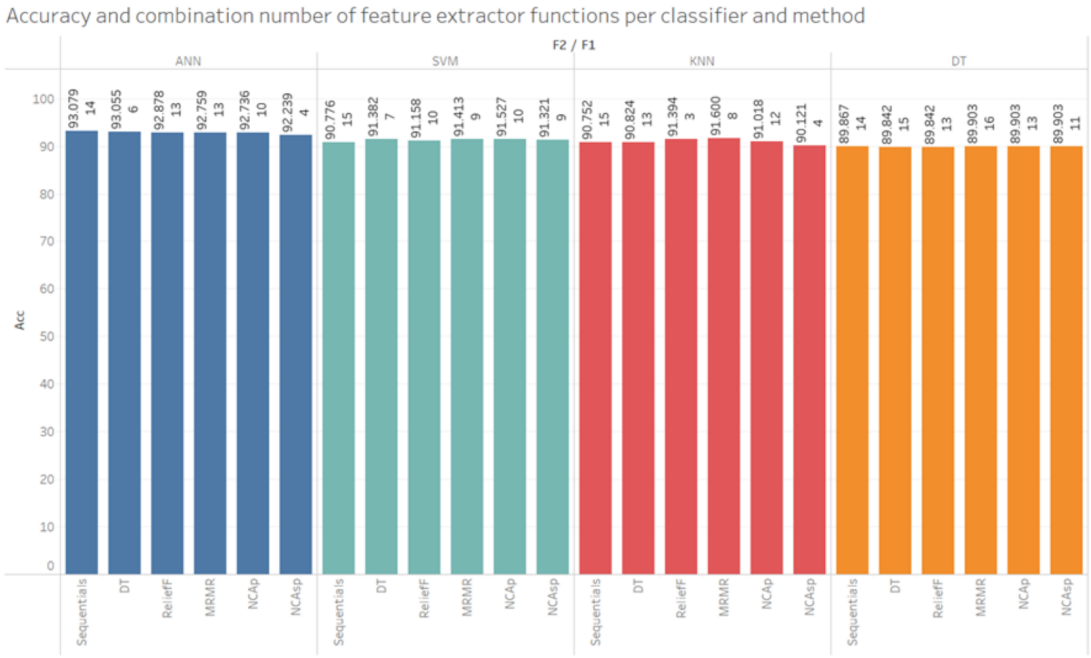

Figure 4.9: Evaluation of the accuracy of the feature selection methods, with the number of combinations of features grouped by the classification algorithm

In table 4.7 illustrates the results of a statistical analysis aimed at discerning significant differences among the various feature selection methods evaluated within the ANN framework. The statistical examination reveals a lack of statistically significant differences between the Sequential and DT methods, despite their status as the most accurate options. However, it's noteworthy that while both methods achieve comparable levels of accuracy, Sequential relies on a combination of 14 feature selector functions, whereas DT utilizes a more streamlined set of 6 functions. This discrepancy prompts a deeper investigation into the response times exhibited by each algorithm when coupled with the respective feature selection methods. Understanding the interplay between accuracy, computational efficiency, and the number of feature extraction functions employed is crucial for optimizing the overall performance of the machine learning model in practical applications.

Methods	diff	lower	upper	p adj
MRMR-DT	-0.272	-0.467	-0.076	0.002
NCAp-DT	-0.295	-0.490	-0.099	0.001
NCAsp-DT	-0.790	-0.995	-0.587	0.000
ReliefF-DT	-0.153	-0.348	0.042	0.204
Sequential-DT	0.048	-0.170	0.270	0.985
NCAp-MRMR	-0.023	-0.192	0.146	0.998
NCAsp-MRMR	-0.592	-0.670	-0.340	0.000
ReliefF-MRMR	0.119	-0.050	0.288	0.309
Sequential-MRMR	0.320	0.125	0.515	0.000
NCAsp-NCAp	-0.496	-0.676	-0.317	0.000
ReliefF-NCAp	0.142	-0.027	0.311	0.146
Sequential-NCAp	0.343	0.148	0.538	0.000
ReliefF-NCAsp	0.638	0.459	0.817	0.000
Sequential-NCAsp	0.839	0.635	1.043	0.000
Sequential-ReliefF	0.201	0.006	0.396	0.040

Table 4.7: Evaluation of the accuracy of the feature selection methods, with the number of combinations of features grouped by the classification algorithm.

In the context of hand gesture recognition utilizing infrared signals from the Leap Motion Controller, the processing time emerges as a pivotal factor influencing the selection of an optimal feature selection method. Illustrated in Figure 4.10, the processing time in milliseconds for feature selection methods at their peak accuracy is depicted. Notably, the Decision Tree (DT) stands out as the feature selection algorithm boasting the shortest processing time, achieving maximum model accuracy with the utilization of just 6 feature extraction functions. This efficiency is particularly advantageous in real-time applications where swift processing is imperative for seamless interaction and response. However, it's crucial to strike a balance between processing time and accuracy, as more complex feature selection methods may offer enhanced precision at the cost of increased computational overhead. Thus, selecting an appropriate feature selection strategy hinges on weighing these factors against the specific requirements and constraints of the application scenario.

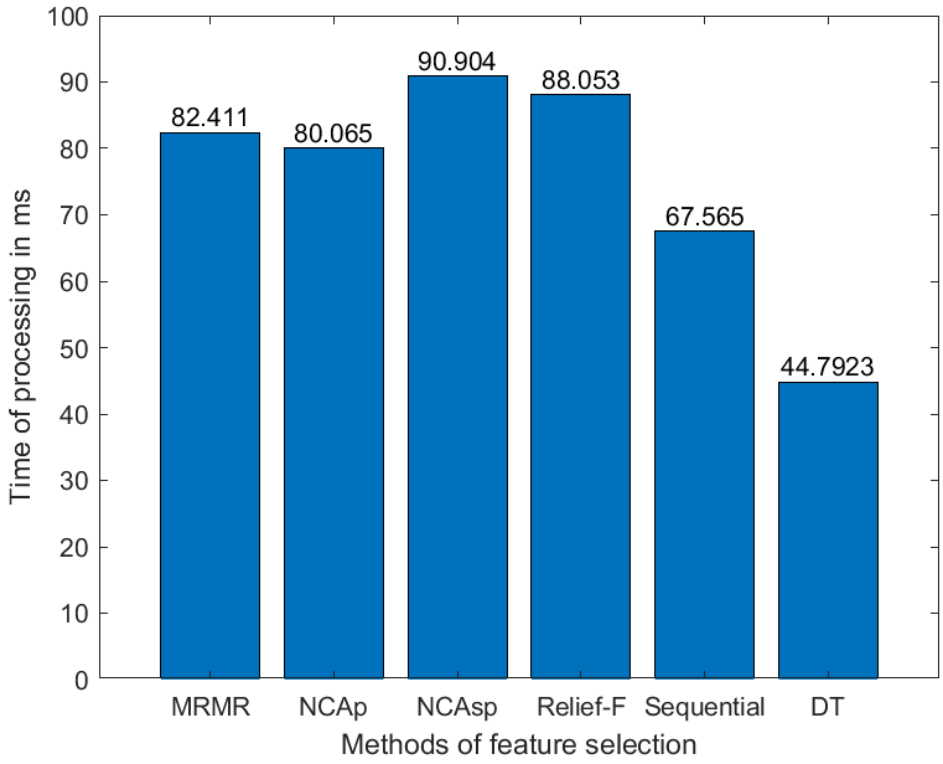

Figure 4.10: Evaluation of the processing time of the feature selection methods.

4.7. Conclusions

This chapter also analyzes the MRMR, Sequential, NCAp, NCAsp, Relief-F, and DT feature selection methods. In addition, we define a hand gesture recognition and classification model to evaluate the score reported by the above feature selection methods. The model comprises five modules: data acquisition, preprocessing, feature extraction, classification, and postprocessing. We test the ANN, SVM, KNN, and DT algorithms in the classification module. The dataset used in this study consists of five gestures. These gestures are open hand, fist, wave in, wave out, and pinch. The dataset contains 1680 observations for each gesture, totaling 8400. The input data is an array of 8400x17. The seventeen predictors are formed by computing the functions MAV, EMAV, MMAV, MMAV2, VAR, RMS, DASDV, SD, MV, ACC, WL EWL, LD, SSC, MYOP, WA, SSI. The output of these functions is fed into the MRMR, Sequential, NCAp, NCAsp, Relief-F and DT feature selection methods. These methods return a score according to the relevance of the feature extraction functions.

The feature selection methods return distinct orders of feature extraction functions based on their scores. MRMR ranks the functions combining the top four functions yields a maximum accuracy of 92.71% for classification and 82.1% for recognition. Sequential method

ranks features differently, present a combination of 14 functions achieves 93.0788% accuracy for testing and 83.5152% for recognition, with a processing time of 67.5646 seconds. NCAp ranks the top ten functions yield 92.7% accuracy for classification and 82.2% for recognition. NCAsp ranks the top four functions to achieve 92.3% accuracy for classification and 81.4% for recognition. Relief-F ranks using thirteen functions yield 92.9% accuracy for test classification and 83.6% for recognition. Finally, DT ranks features combining six functions achieves 93.0545% accuracy for testing and 83.2273% for recognition, with a processing time of 44.7923 seconds.

It's important to note that there's no statistically significant difference between the Sequential and DT methods, which are quite comparable. However, there's a notable discrepancy between Sequential and ReliefF, as well as DT and ReliefF, with Sequential and DT emerging as the top-performing methods. The other approaches exhibit lower accuracy rates. Nonetheless, for our specific problem, employing DT as a feature selection method alongside ANN is recommended. While Sequential performs similarly, DT requires only six functions to achieve optimal accuracy, resulting in significantly reduced processing time.

Chapter 5

Hand gesture recognition using automatic feature extraction and deep learning algorithms with memory

Contents

5.1. Abstract

Gesture recognition serves a crucial role in expressing emotions and facilitating communication between individuals and machines. Hand gesture recognition, in particular, garners significant interest among researchers due to its inherently complex nature as a high-dimensional pattern recognition problem. The challenge lies in effectively capturing and

processing the vast array of data inherent in hand movements. Feature selection and extraction are key strategies to mitigate the dimensionality issue, offering avenues to enhance the performance of machine learning models. To this end, we propose evaluating automatic feature extraction functions and comparing models employing both manual and automatic feature extraction techniques. Manual extraction employs statistical functions to capture central tendencies, while automatic extraction leverages advanced methods such as Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (BiLSTM). These features are further evaluated within classifiers such as Softmax, Artificial Neural Networks (ANN), and Support Vector Machines (SVM). Notably, the most successful model emerges as the combination of BiLSTM and ANN (BiLSTM-ANN), achieving an impressive accuracy rate of 99.9912%.

5.2. Introduction

The quality and quantity of data are closely related to the performance and generalization of machine learning models. The quantity of data depends on the nature of the problem, the technology used to acquire the data, and the availability of the data. In general, machine learning models tend to perform better when trained on larger amounts of data. With more data relative to the model complexity, the model has a greater opportunity to learn the underlying patterns and relationships in the data, which can lead to better predictions and generalization to new, and unseen data. Data quality means that the data is free of errors, noise, and bias, which can improve the accuracy and reliability of machine learning models. Also, data quality is related to how well the features describe the problem [123]–[125].

In this sense, machine learning models are affected by the problem of the curse of dimensionality. This problem occurs when many features are included as inputs of machine learning models. The machine learning models can work in a scenario that is close to falling into an overfitting problem. Overfitting occurs when there are many features relative to the amount of training data. Including many features can also increase the computational complexity of the model, making it more difficult to train and use in practice [76], [126], [127].

It is necessary to use dimensionality reduction techniques. These techniques are feature selection and feature extraction [99]. Feature selection is the selection of the best functions that can represent the problem. At the same time, feature extraction selects and transforms the most relevant and informative features from a data set. Feature extraction is a critical step in machine learning models [128]. This is because the quality and relevance of the

features can greatly affect the performance and accuracy of the model. In many cases, the original data may contain many redundant, noisy, or irrelevant features for the task at hand, leading to overfitting and poor generalization performance. To avoid these problems, it is important to carefully select and preprocess the most relevant and informative features for a given task. The process of feature extraction involves domain knowledge, statistical analysis, and data visualization techniques to identify the most important features and discard the redundant or irrelevant ones. There are two ways to approach the process of obtaining the best features. The first is manual feature extraction (addressed in the previous chapter), and the second is automatic feature extraction.

Manual feature extraction for hand gesture recognition requires domain knowledge and expertise in signal processing and feature engineering. It can be time-consuming and labor-intensive and may require iterative experimentation and refinement to identify the most informative features for a given application. However, it can also be more interpretable and understandable than automatic feature extraction techniques because the features are explicitly defined and selected based on human insight and intuition.

In contrast to automatic feature extraction is a machine-learning technique that involves deep learning [129]. Automatic feature extraction for hand gesture recognition using the Leap Motion Controller could use CNN or Recurrent Neural Networks (RNN) [60], [130]. In addition, automatic feature extraction for hand gesture recognition can save time and effort in feature engineering and can potentially discover more informative and complex features than manual methods. However, it requires large amounts of training data and computational resources and may be more difficult to interpret and understand than manual methods.

In this chapter, we propose an HGR model that takes as input time series based on the spatial positions and directions of the fingers of the hand. In addition, we to comparate manual and automatic feature extraction for the problem. The manual extraction is performed using statistical features such as: pulse percentage rate (MYOP), detector log (LD), wavelength (WL), enhanced wavelength (EWL), difference absolute standard deviation value (DASDV), and standard deviation (SD). These feature extraction functions are selected based on their great performance demonstrated in the previous chapter's evaluation. We use CNN and BiLSTM to extract features automatically. In addition, the extracted features are evaluated using Softmax, ANN, and SVM classifiers. We also propose to evaluate the data on classifiers like CNN-ANN, CNN-SVM, BiLSTM-ANN, and BiLSTM-SVM.

5.3. Description of methods used for developing automatic feature extraction

This subsection uses the spatial positions and directions retrieved by the LMC. The LMC represents the position of the fingertips at time *t* using the matrix $\mathbf{P}_t = [p^{(x)}_{(1,t)}, p^{(y)}_{(1,t)}, p^{(z)}_{(1,t)}; ...; p^{(x)}_{(5,t)}, p^{(y)}_{(5,t)}, p^{(z)}_{(5,t)}]^{(leap)}_t$, being $[p^{(x)}_{(i,t)}, p^{(y)}_{(i,t)}, p^{(z)}_{(i,t)}]$ the vector with the spatial positions of the ith finger with respect to the sensor coordinate axes. The directions of the fingertips at time t are represented using the matrix: $\mathbf{D}_t = [d^{(x)}_{(1,t)}, d^{(y)}_{(1,t)}, d^{(z)}_{(1,t)}; ...; d^{(x)}_{(5,t)}, d^{(y)}_{(5,t)}, d^{(z)}_{(5,t)}]^{(leap)}_t$, being $[d^{(x)}_{(i,t)}, d^{(y)}_{(i,t)}, d^{(z)}_{(i,t)}]$ the vector with the directions of the ith finger with respect to the sensor coordinate axes. For this work, we use the data of the tips of each finger. Then, at each time t, we obtain two matrices $\mathbf{P}_t$ and $\mathbf{D}_t$. The change of the values of the matrices at the times $t_1, t_2, t_3, ..., t_n$ characterizes the hand gesture. This data are processed by statistical functions that manually extract features, and by deep learning algorithms to extract features automatically. In both cases, the features obtained are evaluated in classifiers such as ANN and SVM, as shown in Figure 5.1.

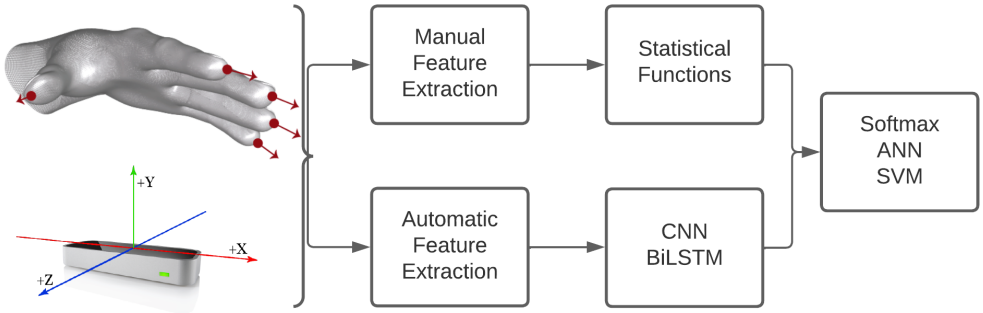

Figure 5.1: Overview of the chapter, where automatic feature extraction is evaluated and compared to manual feature extraction.

5.3.1. Convolutional neural network

CNN is a neural network that has multiple convolutional layers. According to [131], a convolutional layer is a small logistic regression where the convolution mask determines the weights, and the input data values define the constants. The mask can be a matrix or a vector. It is a matrix if the input data is two-dimensional (2D) and a vector if it is one-dimensional

(1D). The output of the convolutional layer can be the same size as the input data if an artifice called padding is used and the step is one. Padding consists of adding values of -1 or 0 outside the size of the input data; the number of values added depends on the size of the mask. A stride is the number of steps in which the convolution is performed. In addition, the CNN has a pooling layer. This layer reduces the dimensionality of the input data. Finally, CNN returns a vector smaller than the input data with a good abstraction or representation of the input data. Figure 5.2 shows the general process of a CNN.

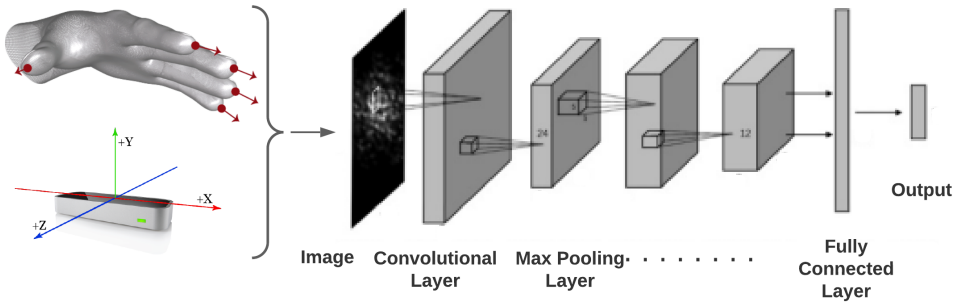

Figure 5.2: Convolutional neural network

5.3.2. Recurrent neural networks

A recurrent neural network (RNN) is designed to perform prediction or classification based on sequential data. These data can be, for example, time series or words within a sentence. In time series, the values of time *t* are inferred from the values of time $t-1$. In the same sense, the values of time $t+1$ are inferred from the values of time *t*. In this context, the challenge for RNNs lies in their ability to process inputs sequentially, ensuring that information at time t is processed in the correct sequence as it was presented at previous $t-1$ and subsequent $t+1$ time steps. However, the inputs at different time steps may vary, posing a challenge for RNN architectures. Specifically, RNNs must effectively manage and update their parameters over time to adapt to changing input patterns while preserving the memory of past inputs. This necessitates the construction of RNN architectures with adaptable parameters that can effectively capture temporal dependencies and patterns within sequential data streams. [132]. The types of RNNs are defined by the different architectures of these neural networks. Among the different types of RNNs, we have long short-term memory (LSTM) and gated recurrent unit (GRU). There is also a variant of LSTM called bidirectional long short-term memory (BiLSTM).

RNNs can assume different configurations depending on the problem to be solved. These configurations are Sequence to Sequence, Sequence to Target, Target to Sequence, and Target to Target. The target-to-target configurations are similar to feed-forward neural networks. This type of neural network has no memory. The target-to-sequence configurations correspond to 1 input at a time, and the model receives many targets. The sequence-to-target configuration is when there is one sequence of input data, and the network can predict one output value. Finally, the sequence-to-sequence configuration corresponds when the input is a sequence of data, where each data is a time instant, and the model predicts a sequence of targets with a label for each time instant.

5.4. Analysis of manual and automatic feature extraction

This chapter evaluates and compares *automatic feature extraction* with *manual feature extraction*. Manual feature extraction is performed by combining statistical functions. Automatic feature extraction is performed using a CNN and a combination of BiLSTM networks. In the same context, features are evaluated using Softmax, ANN, and SVM classifiers. In all experiments, we measure the accuracy of classification and recognition of hand gestures. In addition, during the execution of the experiments, the processing time of prediction is measured. Time is an essential variable in the evaluation of recognition algorithms. This is because these algorithms are used in real-time applications.

5.4.1. Manual feature extraction

In this chapter, we use the following feature extraction function: Pulse percentage rate (MYOP), which calculates the average magnitude of the changes between consecutive data points in a signal, detector log (LD) is good at estimating the exerted force, wavelength (WL) can be calculated by simplifying the cumulative length of the waveform summation, Enhanced Wavelength (EWL) is an extension of WL, Difference Absolute Standard Deviation Value (DASDV) is the square root of the average of the difference between the squared adjacent values, and Standard Deviation (SD). These feature extraction functions are selected from the analysis presented in Chapter 4.4. The dataset, comprising spatial and directional positions, undergoes processing using the windowing technique. Within each window, features are extracted by applying the previously outlined feature extraction functions. These extracted features serve as inputs to the classifiers, facilitating the subsequent classification

process.

In this evaluation, we use a feedforward ANN with two hidden layers. The first hidden layer uses ReLU as an activation function with 25 neurons. The second hidden layer uses logsig as an activation function and 15 neurons. The input of ANN is the number of features according to the number of feature selection functions. We apply gradient descend to the cross-entropy between the predictions of the ANN and the true labels to adjust its weights. Additionally, ANN uses 2000 epochs and a regularization factor of 1.0e-1. For the SVM classifier, we used the Gaussian kernel and a scale factor of order 10. Table 5.1 presents the results of evaluating hand gesture testing and recognition using manual feature extraction on the described ANN and SVM architecture. These architectures are used because they are the best architectures according to the evaluation in the previous chapter.

Manual Feature extraction			
Algorithms	**Classification Testing**	**Recognition**	**Average time of classification testing**
ANN	92.936	83.227	57 milliseconds
SVM	91.370	79.863	

Table 5.1: Average Classification testing and recognition for manual feature extraction

5.4.2. Automatic feature extraction

For automatic feature extraction, we use CNNs. The network receives 30 values as time series of 70 observations as input. The 30 values consist of 15 values for each finger's **X, Y**, and **Z** spatial positions, along with 15 values indicating the directions of the tips of each finger. Consequently, the initial tensor is 30 x 70 in size, serving as input to the CNN, configured with a 1D architecture. The 1D architecture is applied to a one-dimensional convolution network. CNN 1D is a type of neural network commonly used for processing sequential data, such as time series or signals, and processes data along a single dimension, typically the time axis. Its ability to capture local patterns and hierarchical features makes it a powerful tool for extracting meaningful representations from one-dimensional data.

CNN-softmax

The **first convolution** block is formed by a convolution layer consisting of 4 filters of 1x3 with a stride of 1. The second layer is a batchNormalizationLayer with a MeanDecay of 0.1. The third layer is a leakyReluLayer with a scale of 0.01, while the fourth layer is an averagePollingLayer with a poolSize of 5, a stride of 1, and a padding of 0. The **second convolution** block consists of 8 filters of 1x3 and a stride 1. The second layer of this second block is a batchNormalizationLayer with a MeanDecay of 0.1. The third layer is a leakyReluLayer with a scale of 0.01. The fourth layer is an averagePollingLayer with a poolSize of 5, a stride of 1, and a padding of 0. The convolution layers are connected to a fullyConnectedLayer with 5 resulting classes. The output of this layer is normalized by a layerNormalizationLayer with an epsilon of 1x10-5. Because it is a multi-class problem, it is connected to a softmaxLayer and finally connected to a classificationLayer. Figure 5.3 shows the schema of the CNN-softmax model.

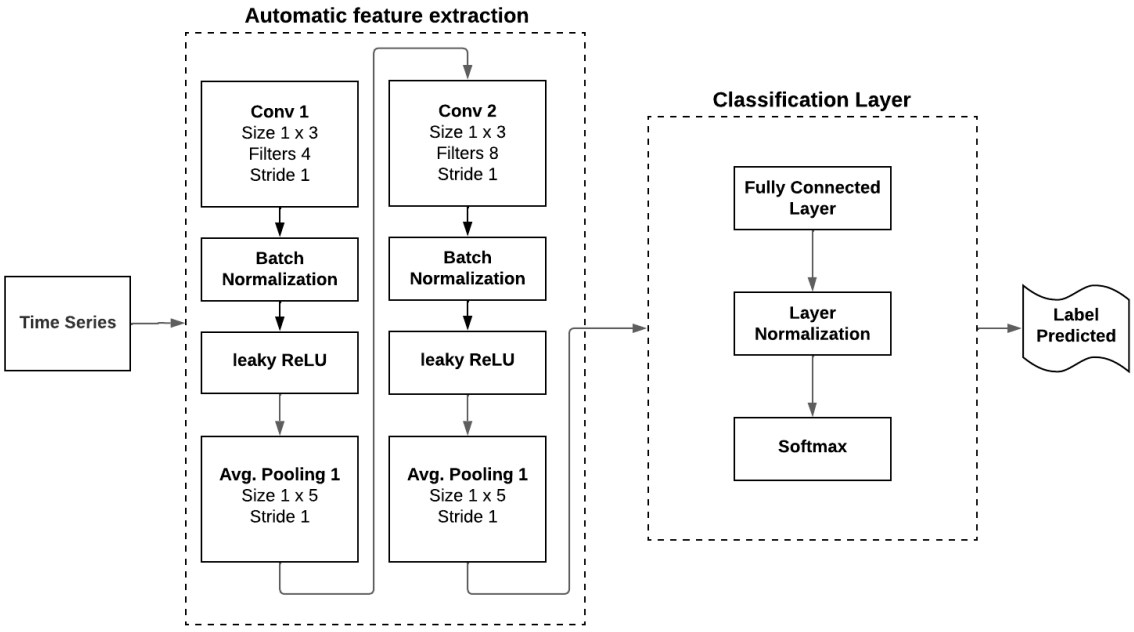

Figure 5.3: Automatic feature extraction using CNN and softmax

The optimization technique for the cost function is stochastic gradient descent with momentum (sgdm). In this work, MATLAB is used to implement the CNN architecture. We set parameters like LearnRateSchedule = piecewise. This parameter allows us to reduce the

learning rate during training, is associated with LearnRateDropFactor = 0.2. This parameter is a multiplicative factor that allows us to reduce the learning rate every certain number of epochs. We set LearnRateDropPeriod = 1, meaning the learning rate is updated at the end of each epoch. Similarly, we set InitialLearnRate = 1e-4 is the value of the initial learning rate. The algorithm is trained for 20 epochs, specified by MaxEpochs = 20. In this scenario, 20 epochs are selected due to the limited training data available, with early stopping and data augmentation utilized to mitigate the risk of overfitting. The training algorithm will group the data into mini-batches to evaluate the gradient of the loss function and update the weights with MiniBatchSize = 32. Finally, we use data augmentation and obtain a dataset three times larger than the original dataset. Table 5.2 shows the results of the classification and recognition of the CNN-softmax model.

CNN-ANN and CNN-SVM

The last layer of the CNN-softmax returns a label. In our case, the model has the layers fullyConnected, layerNormalization, and softmax. The softmax layer reports a label based on the features obtained by the convolutional layer. To implement the CNN-ANN and the CNN-SVM, we excluded the fullyConneted, layerNormalization, and softmax layers from the CNN-softmax model because the particular interest of the work is to obtain the features abstracted by the convolutional blocks. Thus, we enter the CNN architecture at the point where the problem is abstracted and the features are captured before being passed to the fully connectedlayer. This gives us a tensor of 8 x 70 at the end of the CNN. This tensor is flattened, resulting in a feature vector of 560 predictors. This feature vector is fed into the ANN and SVM classification algorithms. The SVM algorithm is trained with a Gaussian kernel and a scale of 10. The architecture of the ANN consists of 1 hidden layer. This hidden layer has 25 neurons, and its activation function is a ReLU function. Also, since the input feature vector is large, a lambda regularization factor of 2.5e-1 is defined. Figure 5.4 shows the automatic feature extraction scheme using CNN with an ANN and an SVM as classifiers.

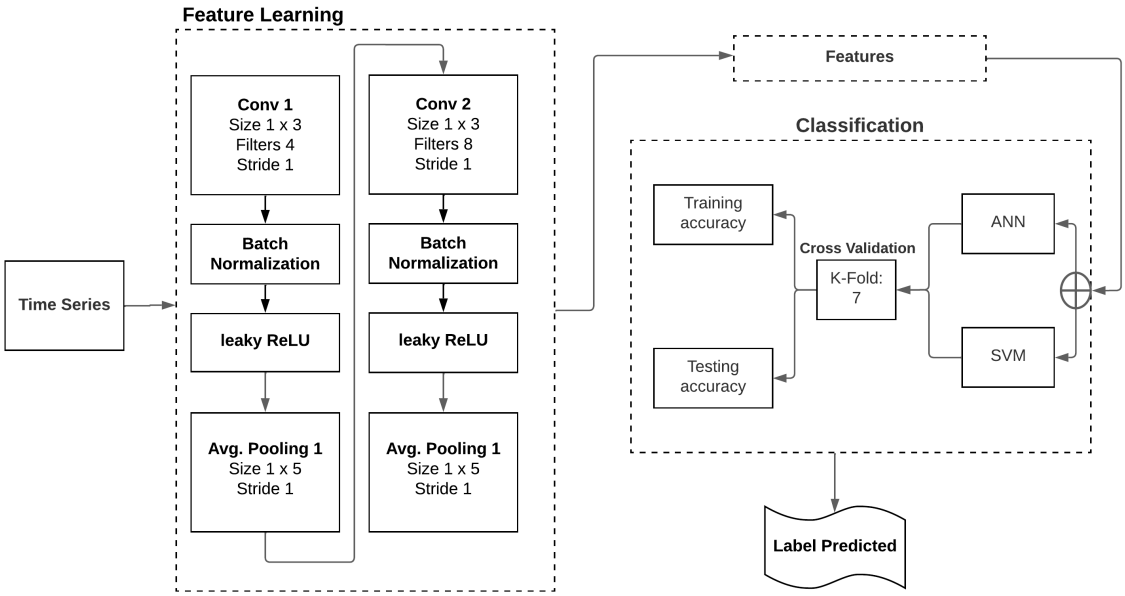

Figure 5.4: Classification using ANN and SVM algorithms with automatic feature extraction by convolution

The ANN and SVM algorithms were trained using the same CNN configuration. Table 5.2 shows the experimental results of automatic feature extraction using CNN and training traditional machine learning algorithms.

Algorithms	**Classification**	**Recognition**	**Average processing time**
CNN-softmax	93.971	88.005	
CNN-ANN	99.795	91.67	30 milliseconds
CNN-SVM	99.403	90.73	

Table 5.2: Accuracy of classification, recognition and around processing time of the CNN-softmax, CNN-ANN, and CNN-SVM model.

In terms of classification accuracy, the CNN-ANN model stands out with the highest accuracy of 99.795%, followed closely by CNN-SVM at 99.403%, and CNN-softmax at 93.971%. This demonstrates the superior performance of CNN-ANN and CNN-SVM over CNN-softmax in classifying hand gestures. Similarly, in recognition accuracy, the CNN-ANN model exhibits the highest performance with an accuracy of 91.67%, followed by CNN-SVM

at 90.73%, and CNN-softmax at 88.005%. This reaffirms the dominance of CNN-ANN in recognizing hand gestures compared to the other models. Notably, the CNN-ANN model boasts a processing time of approximately 30 milliseconds, indicating efficient computational speed. Overall, the CNN-ANN model emerges as the most promising choice among the evaluated models, excelling in both classification and recognition accuracies while maintaining a commendable processing time.

Models with automatic feature extraction and memory cells

We evaluated an RNN of the type BiLSTM in the configuration sequence-to-sequence. This algorithm is fed with the spatial positions and directions of the **X,Y**, and **Z** coordinates of the fingertips. The BiLSTM-ANN and BiLSTM-SVM architecture is shown in Figure 5.5. This algorithm returns a vector of labels, one for each time point. This vector of labels allows us to generate the recognition output.

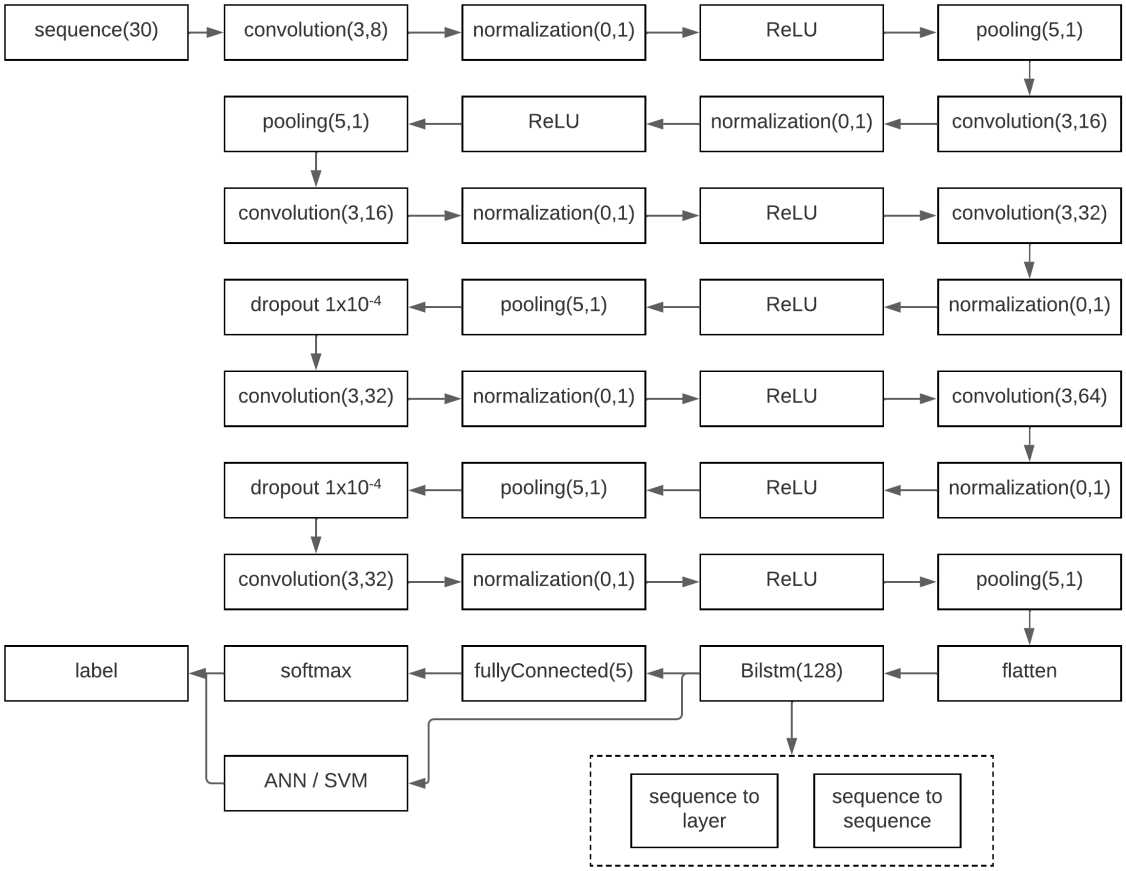

Figure 5.5: Classification using BiLSTM with Softmax, ANN, and SVM algorithms with automatic feature extraction

The BiLSTM has 34 layers: an input sequence layer, 7 convolution layers, 7 normalization layers, 7 ReLU layers, 5 pooling layers, 2 dropout layers, one flatten layer, one BiLSTM, one fully connected layer, one softmax layer, and one classification layer. Figure 5.5 shows each layer's configuration parameters. The convolution layers present a vector of weights or convolution of 1x3 with 8, 16, 32, and 64 filters, respectively These filters are responsible for extracting various features from the input data, capturing different patterns and structures present in the sequential input. In addition, all normalization layers have a normalization factor of 0.1. Similarly, the pooling layers work with the max function with a pool of 5 and jumps of 1. Also, the dropout layer is configured with a regularization factor of 1e-4 to avoid overfitting. Finally, the BiLSTM layer presents 128 gates. Table 5.3 shows the classification and recognition accuracy evaluated in the Softmax classifier.

Furthermore, we obtained the features automatically generated by BiLSTM. These fea-

tures are obtained at the output of the BiLSTM layer. These features were fed to an ANN-based classifier. In the same way, an SVM-based classifier is fed. The ANN and SVM configurations are exactly the same as those used to train the algorithms described in the previous experiments. Table 5.3 shows the training and testing accuracies of BiLSTM-ANN and BiLSTM-SVM.

Algorithms	**Classification**	**Recognition**	**Average processing time**
BiLSTM-softmax	95.616	91.86	
BiLSTM-ANN	99.999	95.73	30 milliseconds
BiLSTM-SVM	99.999	94.79	

Table 5.3: Accuracy of classification, recognition and around processing time of the BiLSTM-softmax, BiLSTM-ANN, and BiLSTM-SVM models.

From table 5.3, it's evident that the BiLSTM-ANN model achieves remarkable results across both classification and recognition tasks. Specifically, it attains a near-perfect classification accuracy of 99.999% and a recognition accuracy of 95.73%. This signifies the robustness and efficacy of the BiLSTM-ANN model in accurately classifying and recognizing hand gestures. Comparatively, the BiLSTM-SVM model also demonstrates excellent performance with classification and recognition accuracies both reaching 99.999% and 94.79%, respectively. However, the BiLSTM-softmax model lags slightly behind, achieving a classification accuracy of 95.616% and a recognition accuracy of 91.86%. Additionally, all three models boast an average processing time of 30 milliseconds, indicating efficient computational speed. Overall, the BiLSTM-ANN and BiLSTM-SVM models emerge as top contenders, showcasing exceptional accuracy in hand gesture classification and recognition tasks.

Figure 5.6 summarizes the accuracies obtained from the manual and automatic feature extraction experiments. It is grouped by feature extraction methods and evaluated in classification algorithms such as ANN, SVM, and softmax. Additionally, the 95% confidence intervals, depicted in gray, are shown to overlap. This does not indicate that there is a significant difference.

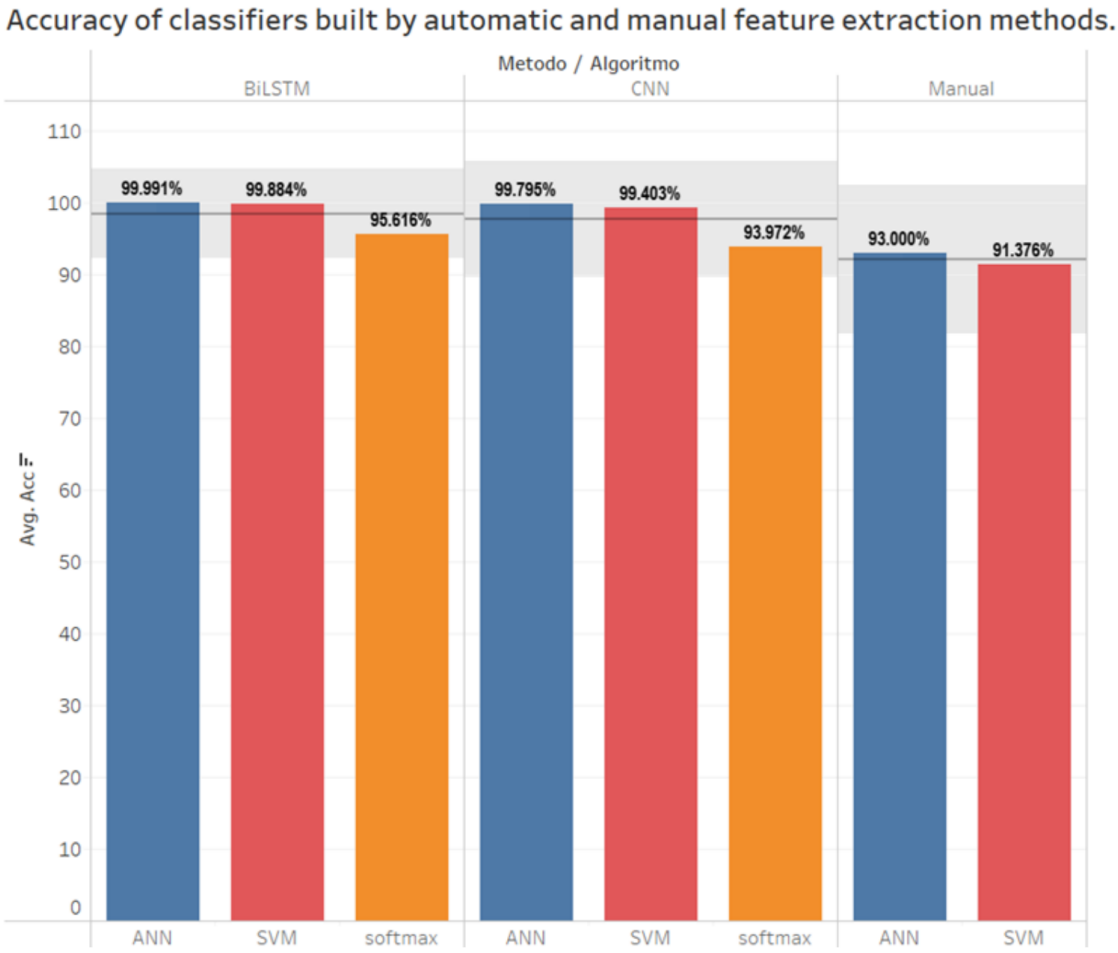

Figure 5.6: Accuracy of ANN, SVM, and Softmax classifiers with automatic feature extraction by CNN and BiLSTM. Accuracy of ANN and SVM classifiers with manual feature extraction

In this context, we generate a statistic to determine if there are significant differences. First, a Shapiro-Wilk normality test is performed, which confirms that the p-value is less than 0.05. Next, a homogeneity of variances test is performed to confirm that the p-value is less than 0.05. Finally, an ANOVA is performed to determine if there are significant differences. Table 5.4 shows the statistical analysis to determine if there is a significant difference.

Table 5.4 in the first column shows the feature extraction methods that are compared to determine if there are significant differences. The column diff represents the difference between the means of two or more groups being compared, lwr Indicates the lower bound of the confidence interval for the difference between group means, upr Denotes the upper bound of the confidence interval for the difference between group means. and 'p adj' refers to the adjusted p-value, which is the probability of observing the test statistic or more ex-

	diff	lwr	upr	p adj
CNN-BiLSTM	-0.1964	-0.3639	-0.0288	0.0205
Manual-BiLSTM	-6.9910	-7.1654	-6.8166	0.0000
Manual-CNN	-6.7946	-6.9690	-6.6202	0.0000

Table 5.4: Pairwise test to determine the method that has a significant difference

treme results, given that the null hypothesis is true. To determine that there is a significant difference in the comparison, the value must be less than 0.05.

For pairwise analysis, the results were filtered to focus on ANN, given its highest reported accuracy. The hypothesis contrast statistic revealed a significant difference between BiLSTM-ANN and CNN-ANN, with BiLSTM-ANN emerging as superior. Furthermore, distinctions were observed between the automatic and manual feature extraction methods, underscoring the importance of method selection in achieving optimal results.

5.5. Conclusion

In this chapter, we have presented the evaluation of 3 hand gesture recognition models, focusing on performance metrics such as classification accuracy, gesture recognition capabilities, and processing time. The data used to evaluate the model were acquired by the LMC. Both accuracy and processing time were measured on the model with manual feature extraction and the models with automatic feature extraction. Manual feature extraction is performed by applying statistical functions of central tendency such as MYOP, LD, WL, EWL, DASDV, and SD. These functions were chosen based on a the previous chapter, where two classifiers, ANN and SVM, are evaluated. The results obtained are 92.936% for ANN and 91.370% for SVM for testing, while for recognition, 83.227% for ANN and 79.863% for SVM. Also, the average classification test time is reported, the test of the models returns about 57 milliseconds. This time is taken over the test set, from when an instance feeds the classifier to when a label is returned. This evaluation shows that the model is running in real-time.

For automatic feature extraction, a CNN and a BiLSTM are used. The CNN is used to obtain an abstraction of the 1D input problem based on convolutions. The 1D input feature vector consists of 2100 features. A total of 516 features were obtained after automatic feature extraction. These features were fed to classic classifiers. The classifiers used are Softmax, ANN, and SVM. For the softmax classifier, the accuracies obtained are 93.971% for classification and 88.005% for recognition. For the evaluation of the features extracted

by the CNN and classified with an ANN (CNN-ANN), an accuracy of 99.795% is obtained. In the same context, the CNN-SVM evaluation reports a 99.403% of accuracy. The average processing time reported for the classification test with CNN-ANN and CNN-SVM is around the 30 milliseconds.

A BiLSTM is used because the data obtained for gesture recognition is a time series. The configuration of BiLSTM is sequence to sequence. This is because the algorithm returns a label for each time point, and this is used for recognition. The BiLSTM evaluated on a softmax classifier gives a classification accuracy of 95.6161%, and a recognition accuracy of 91.8601%. The same features extracted by BiLSTM are evaluated an ANN, BiLSTM-ANN, obtaining an accuracy of 99.9912%, and with a SVM BiLSTM-SVM, obtaining an accuracy of 99.8840%. The processing time in the classification test for the BiLSTM-ANN algorithm and for BiLSTM-SVM is evaluated, with an average processing time around the 27 milliseconds.

After the evaluation of the models, simple models with manual feature extraction and complex models with automatic feature extraction. It is observed that there is a significant difference in the classification accuracy between the simple models and the complex models. The simple model shows a 92.936% of classification accuracy, and the complex model shows a 99.8840%. But between the two complex models CNN-ANN and BiLSTM-ANN, the difference is 0.1962%, which is an almost negligible difference.

Chapter 6

Conclusions and future works

Contents

In this concluding chapter, we synthesize the findings from the extensive exploration of the HGR problem undertaken in this thesis. Additionally, we outline potential avenues for future research and development, aiming to propel the field forward and address lingering gaps and limitations.

6.1. Conclusions

This thesis delves into the issue of hand gesture recognition, which involves complex challenges related to extracting features and recognizing patterns. In order to tackle this research problem, an SLR was undertaken. The analysis of this review reveals that researchers employ different architectures of the models to address hand gesture recognition problems. To establish these architectures, a generic model consisting of modules such as data acquisition, preprocessing, feature extraction, classification, and post-processing serves as a foundational framework. Among the findings, it was noted that the majority of researchers incorporated modules for data acquisition, preprocessing, feature extraction, and classification into their models. Spatial position data emerged as a commonly utilized data type, although a significant portion of studies did not specify the data utilized. Preprocessing techniques encompassed data normalization, noise reduction filters, and segmentation methods, while feature extraction predominantly relied on manual techniques such as statistical methods and finger distance calculations. In terms of classifiers, classical machine

learning algorithms were prevalent, with SVM emerging as the most commonly utilized classifier. Notably, only a single article reported employing post-processing techniques. The primary devices utilized for data acquisition were the Kinect and LMC, with datasets constructed using the LMC typically comprising a limited number of users. Furthermore, all studies employed supervised learning approaches and reported classification accuracies. This helped to identify several gaps and critical areas for future research in the HGR domain, such as: deficiency in data acquisition protocols, data sets created with too few users, absence of recognition algorithms, and decreased model performance due to too many feature extraction functions.

This thesis also includes the creation of a dataset, which was achieved using the LMC involving the participation of 56 individuals. Within this dataset, 9 distinct gestures were defined, comprising 5 static and 4 dynamic gestures. Each participant executed each gesture 30 times, resulting in a dataset containing a total of 15,120 samples, with 8,400 static samples and 6,720 dynamic samples. As identified in the SLR, this dataset is noted for having the largest number of samples among datasets utilized in the realm of hand gesture recognition employing the LMC. Notably, the acquired data encompasses spatial positions, orientations, and images. Additionally, it was determined that the dataset was meticulously constructed to maintain a proportional representation between male and female participants while also ensuring that the age distribution of participants adheres to a normal distribution.

This thesis also analyzes and evaluates the behavior of the manual feature selection and feature extraction methods evaluated in classical machine learning algorithms as ANN, SVM, kNN, and DT. The feature selection methods evaluated are: MRMR, Sequential, NCAp, NCAsp, Relief-F, and DT as feature selection method. The feature extraction methods are: MAV, EMAV, MMAV, MMAV2, VAR, RMS, DASDV, SD, MV, ACC, WL EWL, LD, SSC, MYOP, WA, SSI. The feature extraction methods include statistical functions of central tendency and dispersion. The feature selection methods evaluate and assign scores based on the most effective feature extraction technique for the hand gesture recognition challenge. These scores guide the extraction of features, subsequently inputted into the classifier. Across all feature selection functions, it is consistently observed that an ANN achieves the highest accuracy. Specifically, the sequential method combines 14 feature extraction functions, yielding an accuracy of 93.0788% for testing and 83.5152% for recognition, with a processing time of 67.5646 seconds. Conversely, utilizing a set of 6 feature extraction functions, the

Decision Tree (DT) method achieves an accuracy of 93.0545% for testing and 83.2273% for recognition, with a shorter processing time of 44.7923 seconds.
To discern potential differences, an ANOVA hypothesis test is conducted, revealing that both sequential and DT methods do not exhibit significant variations in reported accuracy. However, a noticeable distinction in processing time is observed, primarily attributed to the disparity in the number of feature extraction functions employed. Given the specifics of our problem domain, it is advisable to employ DT as a feature selection method in conjunction with ANN for optimal performance of the hand gesture recognition model.

In this thesis, we comprehensively assess the performance of automatic feature extraction, focusing on key metrics including classification accuracy, recognition accuracy, and processing time. Employing CNN and BiLSTM models for feature extraction, the extracted features are subsequently fed into classifiers such as Softmax, ANN, and SVM.
For the CNN-based classifiers, notable accuracies are achieved across the board. The CNN-softmax classifier yields 93.971% accuracy for classification and 88.005% for recognition. Evaluating features extracted by CNN and assessed in CNN-ANN results in 99.795% accuracy for classification and 91.67% for recognition, while CNN-SVM reports 99.403% for classification and 90.73% for recognition. The average processing time for classification tests with CNN-softmax, CNN-ANN, and CNN-SVM is approximately 30 milliseconds.
Similarly, the evaluation of BiLSTM-based models showcases results when assessed with a softmax classifier, achieving 95.6161% classification accuracy and 91.8601% recognition accuracy. Transitioning to more complex models, BiLSTM-ANN demonstrates exceptional performance with 99.9912% classification accuracy and 95.73% recognition accuracy, while BiLSTM-SVM records 99.840% for classification and 94.79% for recognition. Notably, the average processing time for classification tests with BiLSTM-Softmax, BiLSTM-ANN, and BiLSTM-SVM is evaluated to be around 30 milliseconds.

A significant disparity in classification accuracy becomes evident upon contrasting classical models with manual feature extraction and deep models with automatic feature extraction. While the classical models achieve 92.936% accuracy, the deep models notably outperform them, showcasing a remarkable 99.999% accuracy. Notwithstanding, the difference between the two deep models, CNN-ANN and BiLSTM-ANN, is a mere 0.1962%, which is deemed negligible in practical terms. Finally, in hand gesture recognition, real-time denotes a system's ability to process and analyze input data, delivering a response within

a timeframe of less than 300 milliseconds. Within the scope of this thesis, the processing times recorded by both manual and automatic feature extraction models fall comfortably under this threshold. Hence, they are deemed to operate effectively in real-time scenarios.

6.2. Future works

Based on the progress made in this thesis, there are exciting opportunities for future research in the HGR field. A future work could be exploring models based on transformer neural networks. These models could evaluate the accuracy of classification and recognition in terms of processing time. Comparing these new transformer-based models with the ones studied in this thesis could offer useful insights into how they perform and what benefits they might offer. This line of research could have the potential to push forward the field of hand gesture recognition, leading to stronger and more effective models.

Furthermore, another field that can be explored is how different data types could be combined to construct datasets for the HGR problem. For example, infrared imaging, electroencephalography (EEG), and electromyography (EMG) could enrich the information available for gesture recognition. Combining these data types could provide complementary information, enhancing the discriminative power of HGR systems and improving their performance in complex real-world scenarios.

Chapter 7

Bibliographic References

[1] N. T. Viet Tuyen and O. Celiktutan, "Agree or Disagree? Generating Body Gestures from Affective Contextual Cues during Dyadic Interactions," in *2022 31st IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, 2022, pp. 1542–1547. DOI: 10.1109/RO-MAN53752.2022.9900760.

[2] N. Harish and S. Poonguzhali, "Design and development of hand gesture recognition system for speech impaired people," *2015 International Conference on Industrial Instrumentation and Control, ICIC 2015*, no. Icic, pp. 1129–1133, 2015. DOI: 10.1109/IIC.2015.7150917.

[3] H. Asutosha, "Gesture to Speech Using Leap Motion Controller," pp. 626–630, 2017.

[4] M. S. Verdadero and J. C. Dela Cruz, "An Assistive Hand Glove for Hearing and Speech Impaired Persons," *2019 IEEE 11th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management, HNICEM 2019*, pp. 1–6, 2019. DOI: 10.1109/HNICEM48295.2019.9072695.

[5] R. McConkey, I. Morris, and M. Purcell, "Communications between staff and adults with intellectual disabilities in naturally occurring settings," *Journal of Intellectual Disability Research*, vol. 43, no. 3, pp. 194–205, 1999, ISSN: 09642633. DOI: 10.1046/j.1365-2788.1999.00191.x.

[6] A. N. Ramesh, C. Kambhampati, J. R. Monson, and P. J. Drew, "Artificial intelligence in medicine," *Annals of the Royal College of Surgeons of England*, vol. 86, no. 5, pp. 334–338, 2004, ISSN: 00358843. DOI: 10.1308/147870804290.

[7] A. Holzinger, G. Langs, H. Denk, K. Zatloukal, and H. Müller, “Causability and explainability of artificial intelligence in medicine,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 9, no. 4, pp. 1–13, 2019, ISSN: 19424795. DOI: 10.1002/widm.1312.

[8] M. Luck and R. Aylett, “Applying artificial intelligence to virtual reality: intelligent virtual environments,” *Applied Artificial Intelligence*, vol. 14, no. 1, pp. 3–32, 2000, ISSN: 10876545. DOI: 10.1080/088395100117142.

[9] U. G. Longo, S. De Salvatore, V. Candela, *et al.*, “Augmented reality, virtual reality and artificial intelligence in orthopedic surgery: A systematic review,” *Applied Sciences (Switzerland)*, vol. 11, no. 7, 2021, ISSN: 20763417. DOI: 10.3390/app11073253.

[10] B. S. Parton, “Sign language recognition and translation: A multidisciplined approach from the field of artificial intelligence,” *Journal of Deaf Studies and Deaf Education*, vol. 11, no. 1, pp. 94–101, 2006, ISSN: 10814159. DOI: 10.1093/deafed/enj003.

[11] I. Papastratis, C. Chatzikonstantinou, D. Konstantinidis, K. Dimitropoulos, and P. Daras, “Artificial intelligence technologies for sign language,” *Sensors*, vol. 21, no. 17, 2021, ISSN: 14248220. DOI: 10.3390/s21175843.

[12] W. Bauer and C. Vocke, *Work in the Age of Artificial Intelligence – Challenges and Potentials for the Design of New Forms of Human-Machine Interaction*. Springer International Publishing, 2020, vol. 961, pp. 493–501, ISBN: 9783030201531. DOI: 10.1007/978-3-030-20154-8_45. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-20154-8_45.

[13] G. Qiao-Franco and I. Bode, “Weaponised Artificial Intelligence and Chinese Practices of Human–Machine Interaction,” *The Chinese Journal of International Politics*, vol. 16, no. 1, pp. 106–128, 2023, ISSN: 1750-8916. DOI: 10.1093/cjip/poac024.

[14] P. K. Pisharady and M. Saerbeck, “Recent methods and databases in vision-based hand gesture recognition : A review,” *Comput. Vis. Image Underst.*, vol. 141, pp. 152–165, 2015, ISSN: 1077-3142. DOI: 10.1016/j.cviu.2015.08.004. [Online]. Available: http://dx.doi.org/10.1016/j.cviu.2015.08.004.

[15] A. Konar and S. Saha, “Gesture recognition,” *Princ. Tech. Appl. Cham Springer Int. Publ.*, 2018.

[16] Zhou Ren, Jingjing Meng, and Junsong Yuan, “Depth camera based hand gesture recognition and its applications in Human-Computer-Interaction,” *2011 8th Int. Conf. Information, Commun. Signal Process.*, pp. 1–5, 2011. DOI: 10.1109/ICICS.2011.6173545. [Online]. Available: http://ieeexplore.ieee.org/document/6173545/.

[17] H. S. Al-Khalifa, “CHEMOTION: A gesture based chemistry virtual laboratory with leap motion,” *Comput. Appl. Eng. Educ.*, vol. 25, no. 6, pp. 961–976, 2017, ISSN: 10990542. DOI: 10.1002/cae.21848.

[18] D. Zhi, T. E. A. D. Oliveira, V. Prado, and E. M. Petriu, “Teaching a Robot Sign Language using Vision-Based Hand Gesture Recognition,” *2018 IEEE Int. Conf. Comput. Intell. Virtual Environ. Meas. Syst. Appl.*, vol. 3528725544, pp. 1–6, 2018.

[19] D. Naglot, “Real Time Sign Language Recognition using the Leap Motion Controller,” pp. 1–5,

[20] W. Abadi, M. Fezari, and R. Hamdi, “BAG OF VISUALWORDS AND CHI-SQUARED KERNEL SUPPORT VECTOR MACHINE : A WAY TO IMPROVE HAND,” no. 1, 2015.

[21] B. Hisham and A. Hamouda, “Arabic Static and Dynamic Gestures Recognition Using Leap Motion,” 2017. DOI: 10.3844/jcssp.2017.337.354.

[22] L. E. Potter, J. Araullo, and L. Carter, “The Leap Motion controller,” *Proc. 25th Aust. Comput. Interact. Conf. Augment. Appl. Innov. Collab. - OzCHI ’13*, no. December 2015, pp. 175–178, 2013. DOI: 10.1145/2541016.2541072. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2541016.2541072.

[23] Zhou Ren, Jingjing Meng, and Junsong Yuan, “Depth camera based hand gesture recognition and its applications in Human-Computer-Interaction,” *2011 8th Int. Conf. Information, Commun. Signal Process.*, pp. 1–5, 2011. DOI: 10.1109/ICICS.2011.6173545. [Online]. Available: http://ieeexplore.ieee.org/document/6173545/.

[24] G. Chernyshov, C. Caremel, and K. Kunze, “Hand Motion Prediction for Just-in-Time Thermo-Haptic Feedback,” 2018.

[25] C. R. Naguri and R. C. Bunescu, “Recognition of dynamic hand gestures from 3D motion data using LSTM and CNN architectures,” *Proc. - 16th IEEE Int. Conf. Mach. Learn. Appl. ICMLA 2017*, vol. 2018-Janua, pp. 1130–1133, 2018, ISSN: 0022-1899. DOI: 10.1109/ICMLA.2017.00013.

[26] K. Lai and S. N. Yanushkevich, “CNN + RNN Depth and Skeleton based Dynamic Hand Gesture Recognition,” *2018 24th Int. Conf. Pattern Recognit.*, pp. 3451–3456, 2018.

[27] B. Engineering, “Hand Gesture Recognition for Post-stroke Rehabilitation Using Leap Motion,” no. c, pp. 386–388, 2017.

[28] B. M. Napoleão, K. R. Felizardo, É. F. De Souza, and N. L. Vijaykumar, “Practical similarities and differences between Systematic Literature Reviews and Systematic Mappings: A tertiary study,” *Proc. Int. Conf. Softw. Eng. Knowl. Eng. SEKE*, no. June 2018, pp. 85–90, 2017, ISSN: 23259086. DOI: 10.18293/SEKE2017-069.

[29] F. Dominio, M. Donadeo, G. Marin, P. Zanuttigh, and G. M. Cortelazzo, “Hand Gesture Recognition with Depth Data,” *Proc. 4th ACM/IEEE Int. Work. Anal. Retr. Tracked Events Motion Imag. Stream*, pp. 9–16, 2013. DOI: 10.1145/2510650.2510651.

[30] X. Dqj, L. D. R. Dl, H. Q. Xq, *et al.*, “A Novel Feature Extracting Method for Dynamic Gesture Recognition Based on Support Vector Machine,” *Proc. IEEE Int. Conf. Inf. Autom.*, no. July, pp. 437–441, 2014.

[31] H. M. Jais, Z. R. Mahayuddin, and H. Arshad, “A Review on Gesture Recognition Using Kinect,” *5th Int. Conf. Electr. Eng. Informatics 2015*, pp. 594–599, 2015. DOI: 10.1109/ICEEI.2015.7352569.

[32] G. Plouffe and A.-M. Cretu, “Static and Dynamic Hand Gesture Recognition in Depth Data Using Dynamic Time Warping,” *IEEE Trans. Instrum. Meas.*, vol. 65, no. 2, pp. 305–316, 2016, ISSN: 0018-9456. DOI: 10.1109/TIM.2015.2498560. [Online]. Available: http://ieeexplore.ieee.org/document/7332940/.

[33] K. Czuszynski, J. Ruminski, and J. Wtorek, “Pose classification in the gesture recognition using the linear optical sensor,” *Proc. - 2017 10th Int. Conf. Hum. Syst. Interact. HSI 2017*, pp. 18–24, 2017. DOI: 10.1109/HSI.2017.8004989.

[34] Y. Xi, S. Cho, S. Fong, Y. W. Park, and K. Cho, “Gesture Recognition Method Using Sensing Blocks,” *Wirel. Pers. Commun.*, vol. 91, no. 4, pp. 1779–1797, 2016, ISSN: 1572-834X. DOI: 10.1007/s11277-016-3356-z.

[35] P. Jangyodsuk, C. Conly, and V. Athitsos, “Sign language recognition using dynamic time warping and hand shape distance based on histogram of oriented gradient features,” *Proc. 7th Int. Conf. PErvasive Technol. Relat. to Assist. Environ. - PETRA ’14*,

pp. 1–6, 2014. DOI: 10.1145/2674396.2674421. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2674396.2674421.

[36] G. Plouffe and A.-m. Cretu, "Static and Dynamic Hand Gesture Recognition in Depth Data Using Dynamic Time Warping," pp. 1–12, 2015.

[37] H. Doan, H. Vu, and T. Tran, "Recognition of hand gestures from cyclic hand movements using spatial-temporal features," *ACM Int. Conf. Proceeding Ser.*, vol. 03-04-Dece, pp. 260–267, 2015. DOI: 10.1145/2833258.2833301.

[38] W. Lu, Z. Tong, and J. Chu, "Dynamic hand gesture recognition with leap motion controller," *IEEE Signal Process. Lett.*, vol. 23, no. 9, pp. 1188–1192, 2016, ISSN: 10709908. DOI: 10.1109/LSP.2016.2590470.

[39] R. E. Nogales and M. E. Benalcázar, "Hand gesture recognition using machine learning and infrared information: a systematic literature review," *Int. J. Mach. Learn. Cybern.*, 2021, ISSN: 1868-808X. DOI: 10.1007/s13042-021-01372-y. [Online]. Available: https://doi.org/10.1007/s13042-021-01372-y.

[40] M. Benmoussa and A. Mahmoudi, "Machine Learning for Hand Gesture Recognition Using Bag-of-words,"

[41] K. M. Vamsikrishna, D. P. Dogra, and M. S. Desarkar, "Computer-Vision-Assisted Palm Rehabilitation," *IEEE Trans. Biomed. Eng.*, vol. 63, no. 5, pp. 991–1001, 2016. DOI: 10.1109/TBME.2015.2480881.

[42] M. A. Almasre and H. Al-nuaim, "Recognizing Arabic Sign Language Gestures Using Depth Sensors and a KSVM Classifier," pp. 146–151, 2016.

[43] D. Avola, M. Bernardi, L. Cinque, *et al.*, "Exploiting Recurrent Neural Networks and Leap Motion Controller for Sign Language and Semaphoric Gesture Recognition," pp. 1–11, arXiv: arXiv:1803.10435v1.

[44] F. Liu, B. Du, Q. Wang, Y. Wang, and W. Zeng, "Hand Gesture Recognition Using Kinect via Deterministic Learning," pp. 2127–2132, 2017.

[45] A. B. Khalifa, "A Comprehensive Leap Motion Database for Hand Gesture Recognition," no. July 2013, pp. 514–519, 2016.

[46] N. Rossol, I. Cheng, S. Member, A. Basu, and S. Member, "A Multisensor Technique for Gesture Recognition Through Intelligent Skeletal Pose Analysis," *IEEE Trans. Human-Machine Syst.*, vol. 46, no. 3, pp. 350–359, 2016. DOI: 10.1109/THMS.2015.2467212.

[47] A. Clark and D. Moodley, “A System for a Hand Gesture-Manipulated Virtual Reality Environment,” *Proc. Annu. Conf. South African Inst. Comput. Sci. Inf. Technol. - SAICSIT ’16*, pp. 1–10, 2016. DOI: 10.1145/2987491.2987511. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2987491.2987511.

[48] C. Wang, Z. Liu, M. Zhu, J. Zhao, and S.-c. Chan, “A Hand Gesture Recognition System based on Canonical Superpixel-Graph,” *Signal Process. Image Commun.*, 2017, ISSN: 0923-5965. DOI: 10.1016/j.image.2017.06.015. [Online]. Available: http://dx.doi.org/10.1016/j.image.2017.06.015.

[49] W. Liang and L. Guixi, “Dynamic and combined gestures recognition based on multi-feature fusion in a complex environment,” *J. China Univ. Posts Telecommun.*, vol. 22, no. 2, pp. 81–88, 2015, ISSN: 1005-8885. DOI: 10.1016/S1005-8885(15)60643-4. [Online]. Available: http://dx.doi.org/10.1016/S1005-8885(15)60643-4.

[50] X. Bai and C. Li, “Dynamic Hand Gesture Recognition Based On Depth Information,” *2018 Int. Conf. Control. Autom. Inf. Sci.*, no. Iccais, pp. 216–221, 2018.

[51] C. Li, “Hand Gesture Recognition based on Wavelet Invariant Moments,” no. 61403302, pp. 459–464, 2017. DOI: 10.1109/ISM.2017.91.

[52] A. O. Tang, K. E. Lu, Y. Wang, J. I. E. Huang, and H. Li, “A Real-Time Hand Posture Recognition System Using,” vol. 6, no. 2, 2015.

[53] Cheng, “Author ’ s Accepted Manuscript An Image-to-Class Dynamic Time Warping Approach for both 3D Static and Trajectory Hand Gesture Recognition,” *Pattern Recognit.*, 2016, ISSN: 0031-3203. DOI: 10.1016/j.patcog.2016.01.011. [Online]. Available: http://dx.doi.org/10.1016/j.patcog.2016.01.011.

[54] K. M. Vamsikrishna, D. P. Dogra, and M. S. Desarkar, “Computer-Vision-Assisted Palm Rehabilitation With Supervised Learning,” *IEEE Trans. Biomed. Eng.*, vol. 63, no. 5, pp. 991–1001, 2016, ISSN: 15582531. DOI: 10.1109/TBME.2015.2480881.

[55] N. Rossol, I. Cheng, S. Member, A. Basu, and S. Member, “A Multisensor Technique for Gesture Recognition Through Intelligent Skeletal Pose Analysis,” pp. 1–10, 2015.

[56] W. Tao, M. C. Leu, and Z. Yin, “Engineering Applications of Artificial Intelligence American Sign Language alphabet recognition using Convolutional Neural Networks with multiview augmentation and inference fusion,” *Eng. Appl. Artif. Intell.*, vol. 76, no. February, pp. 202–213, 2018, ISSN: 0952-1976. DOI: 10.1016/j.engappai.2018.09.006. [Online]. Available: https://doi.org/10.1016/j.engappai.2018.09.006.

[57] M. A. Almasre and H. Al-Nuaim, “Recognizing Arabic sign language gestures using depth sensors and a KSVM classifier,” *2016 8th Comput. Sci. Electron. Eng. Conf. CEEC 2016 - Conf. Proc.*, pp. 146–151, 2017. DOI: 10.1109/CEEC.2016.7835904.

[58] F. Jiang, S. Zhang, S. Wu, Y. Gao, and D. Zhao, “Multi-layered Gesture Recognition with Kinect,” vol. 16, pp. 227–254, 2015.

[59] D.-y. Hsiao, M. Sun, C. Ballweber, and S. Cooper, “Proactive Sensing for Improving Hand Pose Estimation,” pp. 2348–2352, 2016.

[60] C. R. Naguri and R. C. Bunescu, “Recognition of dynamic hand gestures from 3D motion data using LSTM and CNN architectures,” *Proc. - 16th IEEE Int. Conf. Mach. Learn. Appl. ICMLA 2017*, vol. 2017-Decem, pp. 1130–1133, 2017. DOI: 10.1109/ICMLA.2017.00013.

[61] G. Strezoski, D. Stojanovski, and I. Dimitrovski, “Hand Gesture Recognition Using Deep Convolutional Neural Networks,” 2016.

[62] J. C. Núñez, R. Cabido, J. J. Pantrigo, A. S. Montemayor, and J. F. Vélez, “Convolutional Neural Networks and Long Short-Term Memory for skeleton-based human activity and hand gesture recognition,” vol. 76, pp. 80–94, 2018. DOI: 10.1016/j.patcog.2017.10.033.

[63] M. Benmoussa and A. Mahmoudi, “Machine learning for hand gesture recognition using bag-of-words,” *2018 Int. Conf. Intell. Syst. Comput. Vision, ISCV 2018*, vol. 2018-May, pp. 1–7, 2018. DOI: 10.1109/ISACV.2018.8354082.

[64] C. Hong, Z. Zeng, R. Xie, W. Zhuang, and X. Wang, “Domain adaptation with low-rank alignment for weakly supervised hand pose recovery,” *Signal Processing*, vol. 142, pp. 223–230, 2018, ISSN: 0165-1684. DOI: 10.1016/j.sigpro.2017.07.032. [Online]. Available: http://dx.doi.org/10.1016/j.sigpro.2017.07.032.

[65] W. Zeng, C. Wang, and Q. Wang, “Hand gesture recognition using Leap Motion via deterministic learning,” pp. 28 185–28 206, 2018.

[66] D. Avola, M. Bernardi, L. Cinque, G. L. Foresti, and C. Massaroni, “Exploiting Recurrent Neural Networks and Leap Motion Controller for the Recognition of Sign Language and Semaphoric Hand Gestures,” *IEEE Trans. Multimed.*, vol. 21, no. 1, pp. 234–245, 2019, ISSN: 15209210. DOI: 10.1109/TMM.2018.2856094.

[67] S. Ameur, A. B. Khalifa, and M. S. Bouhlel, “A comprehensive leap motion database for hand gesture recognition,” *2016 7th Int. Conf. Sci. Electron. Technol. Inf. Telecommun. SETIT 2016*, no. July 2013, pp. 514–519, 2017, ISSN: 10591478. DOI: 10.1109/SETIT.2016.7939924.

[68] D.-y. Hsiao, M. Sun, C. Ballweber, and S. Cooper, “Proactive Sensing for Improving Hand Pose Estimation,” pp. 2348–2352, 2016.

[69] E. Nasr-esfahani, N. Karimi, and S. M. R. Soroushmehr, “Hand Gesture Recognition for Contactless Device Control in Operating Rooms,”

[70] J. Yang and R. Horie, “An Improved Computer Interface Comprising a Recurrent Neural Network and a Natural User Interface,” *Procedia - Procedia Comput. Sci.*, vol. 60, pp. 1386–1395, 2015, ISSN: 1877-0509. DOI: 10.1016/j.procs.2015.08.213. [Online]. Available: http://dx.doi.org/10.1016/j.procs.2015.08.213.

[71] X. Jiang, Z. Gang, and X. Carlo, “Virtual grasps recognition using fusion of Leap Motion and force myography,” *Virtual Real.*, vol. 22, no. 4, pp. 297–308, 2018, ISSN: 1434-9957. DOI: 10.1007/s10055-018-0339-2. [Online]. Available: https://doi.org/10.1007/s10055-018-0339-2.

[72] F. Liu, B. Du, Q. Wang, Y. Wang, and W. Zeng, “Hand Gesture Recognition Using Kinect via Deterministic Learning,” pp. 2127–2132, 2017.

[73] C. Li, “Hand Gesture Recognition based on Wavelet Invariant Moments,” no. 61403302, pp. 459–464, 2017. DOI: 10.1109/ISM.2017.91.

[74] Y. Ye and P. Nurmi, “Gestimator - Shape and Stroke Similarity Based Gesture Recognition Categories and Subject Descriptors,” pp. 219–226,

[75] A. O. Tang, K. E. Lu, Y. Wang, J. I. E. Huang, and H. Li, “A Real-Time Hand Posture Recognition System Using,” vol. 6, no. 2, 2015.

[76] Hong, “Author ’ s Accepted Manuscript An Image-to-Class Dynamic Time Warping Approach for both 3D Static and Trajectory Hand Gesture Recognition,” *Pattern Recognit.*, 2016, ISSN: 0031-3203. DOI: 10.1016/j.patcog.2016.01.011. [Online]. Available: http://dx.doi.org/10.1016/j.patcog.2016.01.011.

[77] K. Inoue, T. Shiraishi, M. Yoshioka, and H. Yanagimoto, “Depth Sensor Based Automatic Hand Region Extraction by Using Time-Series Curve and Its Application to Japanese Finger-spelled Sign Language Recognition,” *Procedia - Procedia Comput.*

Sci., vol. 60, pp. 371–380, 2015, ISSN: 1877-0509. DOI: 10.1016/j.procs.2015.08.145. [Online]. Available: http://dx.doi.org/10.1016/j.procs.2015.08.145.

[78] G. C. Lee, F.-h. Yeh, and Y.-h. Hsiao, "Kinect-based Taiwanese sign-language recognition system," no. 151, pp. 261–279, 2016. DOI: 10.1007/s11042-014-2290-x.

[79] S. Ameur, A. Ben Khalifa, and M. S. Bouhlel, "Chronological pattern indexing: An efficient feature extraction method for hand gesture recognition with Leap Motion," *J. Vis. Commun. Image Represent.*, vol. 70, p. 102 842, 2020, ISSN: 10959076. DOI: 10.1016/j.jvcir.2020.102842. [Online]. Available: https://doi.org/10.1016/j.jvcir.2020.102842.

[80] A. Clark, "A System for a Hand Gesture-Manipulated Virtual Reality Environment," 2016.

[81] D. Q. Leite, J. C. Duarte, L. P. Neves, J. C. D. Oliveira, and G. A. Giraldi, "Hand gesture recognition from depth and infrared Kinect data for CAVE applications interaction," pp. 20 423–20 455, 2017. DOI: 10.1007/s11042-016-3959-0.

[82] H. Liu and L. Wang, "International Journal of Industrial Ergonomics Gesture recognition for human-robot collaboration : A review," *Int. J. Ind. Ergon.*, pp. 1–13, 2017, ISSN: 0169-8141. DOI: 10.1016/j.ergon.2017.02.004. [Online]. Available: http://dx.doi.org/10.1016/j.ergon.2017.02.004.

[83] R. Nogales, M. E. Benalcazar, B. Toalumbo, A. Palate, R. Martinez, and J. Vargas, "Construction of a Dataset for Static and Dynamic Hand Tracking Using a Non-invasive Environment," *Adv. Intell. Syst. Comput.*, vol. 1307 AISC, pp. 185–197, 2021, ISSN: 21945365. DOI: 10.1007/978-981-33-4565-2_12.

[84] P. Visconti, F. Gaetani, G. A. Zappatore, and P. Primiceri, "Technical Features and Functionalities of Myo Armband : An Overview on Related Literature and Advanced Applications of Myoelectric Armbands Technical Features and Functionalities of Myo Armband : An Overview on Related Literature and Advanced Applications," no. September, 2018. DOI: 10.21307/ijssis-2018-005.

[85] E. J. R.-r. A. Marin-hernandez and H. V. Rios-figueroa, "A human – computer interface for wrist rehabilitation : a pilot study using commercial sensors to detect wrist movements," *Vis. Comput.*, vol. 35, no. 1, pp. 41–55, 2019, ISSN: 1432-2315. DOI: 10.1007/s00371-017-1446-x.

[86] F. Trujillo-Romero and S. O. Caballero-Morales, “3D data sensing for hand pose recognition,” *23rd Int. Conf. Electron. Commun. Comput. CONIELECOMP 2013*, pp. 109–113, 2013. DOI: 10.1109/CONIELECOMP.2013.6525769.

[87] S. J. Ryu, J. S. Suh, S. H. Baek, S. Hong, and J. H. Kim, “Feature-Based Hand Gesture Recognition Using an FMCW Radar and its Temporal Feature Analysis,” *IEEE Sens. J.*, vol. 18, no. 18, pp. 7593–7602, 2018, ISSN: 1530437X. DOI: 10.1109/JSEN.2018.2859815.

[88] Ultraleap, “Leap Motion Controller The world ’ s leading hand tracking technology,” pp. 15–16, 2019. [Online]. Available: https://www.ultraleap.com/.

[89] B. Raman and P. P. Roy, *International Conference on Computer Vision and Image Processing, CVIP 2016*. 2017, vol. 459 AISC, pp. 1–644, ISBN: 9789811021039.

[90] G. Marin, F. Dominio, and P. Zanuttigh, “Hand gesture recognition with jointly calibrated Leap Motion and depth sensor,” *Multimed. Tools Appl.*, vol. 75, no. 22, pp. 14 991–15 015, 2016, ISSN: 15737721. DOI: 10.1007/s11042-015-2451-6.

[91] D. P. Quintero Lorza, N. D. Duque Méndez, and J. A. Gómez Soto, *GLORIA: A Genetic Algorithms Approach to Tetris*. 2020, vol. 1078, pp. 111–126, ISBN: 9783030336134. DOI: 10.1007/978-3-030-33614-1_8.

[92] F. Weichert, D. Bachmann, B. Rudak, and D. Fisseler, “Analysis of the accuracy and robustness of the Leap Motion Controller,” *Sensors (Switzerland)*, vol. 13, no. 5, pp. 6380–6393, 2013, ISSN: 14248220. DOI: 10.3390/s130506380.

[93] R. Nogales and M. Benalcázar, “Real-Time Hand Gesture Recognition Using the Leap Motion Controller and Machine Learning,” pp. 1–6,

[94] J. Gong, Y. Zhang, X. Zhou, and X.-d. Yang, “Pyro : Thumb-Tip Gesture Recognition Using Pyroelectric Infrared Sensing,” 2017.

[95] R. Wang, F. R. Labs, K. Kin, and F. R. Labs, “Online Optical Marker-based Hand Tracking with Deep Labels,” vol. 37, no. 4, 2018.

[96] S. Raschka, “STAT 479: Machine Learning Lecture Notes,” 2018.

[97] M. E. Benalc, C. E. Anchundia, P. Zambrano, and M. Segura, “A Model for Real-Time Hand Gesture Recognition Using Electromyography (EMG), Covariances and Feed-Forward Artificial Neural Networks,”

[98] H. Tang, W. Wang, D. Xu, Y. Yan, and N. Sebe, “GestureGAN for Hand Gesture-to-Gesture Translation in the Wild,” pp. 774–782, 2018.

[99] N. Normani, A. Urru, L. Abraham, *et al.*, “A Machine Learning Approach for Gesture Recognition with a Lensless Smart Sensor System,” no. March, pp. 4–7, 2018.

[100] M. H. Bataineh, “Artificial neural network for studying human performance,” *ProQuest Diss. Theses*, vol. 1518568, p. 179, 2012. [Online]. Available: http://ezproxy.net.ucf.edu/login?url=http://search.proquest.com/docview/1039557097?accountid=10003%7B%5C%%7D5Cnhttp://sfx.fcla.edu/ucf?url%7B%5C_%7Dver=Z39.88-2004%7B%5C&%7Drft%7B%5C_%7Dval%7B%5C_%7Dfmt=info:ofi/fmt:kev:mtx:dissertation%7B%5C&%7Dgenre=dissertations+%7B%5C&%7D+theses%7B%5C&%7Dsid=ProQ:ProQuest+Dissertations+%7B%5C&%7D+.

[101] S. Alpaydin, R. Gutierrez-osuna, and P. Analysis, “Support-Vector Machines,”

[102] R. H. Bishop, *THE MECHATRONICS*, ISBN: 0849300665.

[103] A. Destrero, S. Mosci, C. De Mol, A. Verri, and F. Odone, “Feature selection for high-dimensional data,” *Comput. Manag. Sci.*, vol. 6, no. 1, pp. 25–40, 2009, ISSN: 1619697X. DOI: 10.1007/s10287-008-0070-7.

[104] K. Y. Fok, N. Ganganath, C. T. Cheng, and C. K. Tse, “A Real-Time ASL Recognition System Using Leap Motion Sensors,” *Proc. - 2015 Int. Conf. Cyber-Enabled Distrib. Comput. Knowl. Discov. CyberC 2015*, pp. 411–414, 2015. DOI: 10.1109/CyberC.2015.81.

[105] I. N. Midarto Dwi Wibowo, “2017 International Conference on Information & Communication Technology and System (ICTS),” *2017 Int. Conf. Inf. Commun. Technol. Syst.*, pp. 67–72, 2017.

[106] E. Guerra-Segura, A. Ortega-Pérez, and C. M. Travieso, “In-air signature verification system using Leap Motion,” *Expert Syst. Appl.*, vol. 165, no. August 2020, 2021, ISSN: 09574174. DOI: 10.1016/j.eswa.2020.113797.

[107] A. Borysova, “Title: Leap Motion Controller for South African Sign Language Recognition,” 2016. [Online]. Available: https://projects.cs.uct.ac.za/honsproj/cgi-bin/view/2017/borysova%7B%5C_%7Dkooverjee%7B%5C_%7Dversfeld.zip/supporting/final%7B%5C_%7Dpaper%7B%5C_%7Danna.pdf.

[108] A. G. Sooai, P. Batarius, Y. C. H. Siki, *et al.*, “Comparison of Recognition Accuracy on Dynamic Hand Gesture Using Feature Selection,” *2018 Int. Conf. Comput. Eng. Netw. Intell. Multimedia, CENIM 2018 - Proceeding*, pp. 270–274, 2018. DOI: 10.1109/CENIM.2018.8711397.

[109] C. M. Bishop, *Neural Networks for Pattern Recognition*. New York: OXFORD university press, 2005, p. 477, ISBN: 0198538642.

[110] K. L. Du and M. N. Swamy, "Neural networks in a softcomputing framework," *Neural Networks a Softcomputing Framew.*, pp. 1–566, 2006. DOI: 10.1007/1-84628-303-5.

[111] I. A. Basheer and M. Hajmeer, "Artificial neural networks: Fundamentals, computing, design, and application," *J. Microbiol. Methods*, vol. 43, no. 1, pp. 3–31, 2000, ISSN: 01677012. DOI: 10.1016/S0167-7012(00)00201-3.

[112] V. Jakkula, "Tutorial on Support Vector Machine (SVM)," *Sch. EECS, Washingt. State Univ.*, pp. 1–13, 2011. [Online]. Available: http://www.ccs.neu.edu/course/cs5100f11/resources/jakkula.pdf.

[113] D. G. Hart, Peterr E. Stork, *Pattern Classification*, second. 1985, vol. 37, pp. 520–524. DOI: 10.1080/09668138508411607.

[114] R. Zhang, Y. Ming, and J. Sun, "Hand gesture recognition with SURF-BOF based on Gray threshold segmentation," *Int. Conf. Signal Process. Proceedings, ICSP*, vol. 0, pp. 118–122, 2016. DOI: 10.1109/ICSP.2016.7877808.

[115] N. N. Algorithms, "Lecture 1 k-Nearest Neighbor Algorithms for Classification and Prediction," pp. 1–6, [Online]. Available: https://ocw.mit.edu/courses/sloan-school-of-management/15-062-data-mining-spring-2003/lecture-notes/knn3.pdf.

[116] D. Science, "K-Nearest Neighbors (KNN) Classification with Different 1 Introduction 2 Method Analysis," pp. 2019–2020, 2020.

[117] S. R. Safavian and D. Landgrebe, "A Survey of Decision Tree Classifier Methodology," *IEEE Trans. Syst. Man Cybern.*, vol. 21, no. 3, pp. 660–674, 1991, ISSN: 21682909. DOI: 10.1109/21.97458.

[118] J. Too, A. R. Abdullah, and N. M. Saad, "Classification of Hand movements based on discrete wavelet transform and enhanced feature extraction," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 6, pp. 83–89, 2019, ISSN: 21565570. DOI: 10.14569/ijacsa.2019.0100612.

[119] L. Yu and H. Liu, "Redundancy based feature selection for microarray data," *KDD-2004 - Proc. Tenth ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, no. 2, pp. 737–742, 2004. DOI: 10.1145/1014052.1014149.

[120] MathWorks, "Introducing Machine Learning What is Machine,"

[121] W. Yang, K. Wang, and W. Zuo, “Neighborhood component feature selection for high-dimensional data,” *J. Comput.*, vol. 7, no. 1, pp. 162–168, 2012, ISSN: 1796203X. DOI: 10.4304/jcp.7.1.161-168.

[122] M. Robnik and I. Konenko, “Theoretical and empirical analysis of ReliefF and RReliefF,” *Mach. Learn.*, vol. 53, no. 1–2, pp. 23–69, 2003.

[123] A. Kariluoto, J. Kultanen, J. Soininen, A. Pärnänen, and P. Abrahamsson, “Quality of Data in Machine Learning,” in *2021 IEEE 21st Int. Conf. Softw. Qual. Reliab. Secur. Companion*, 2021, pp. 216–221. DOI: 10.1109/QRS-C55045.2021.00040.

[124] C. E. Valderrama, B. Martinez, N. Katebi, *et al.*, “Improving the quality of point of care diagnostics with real-time machine learning in low literacy LMIC settings,” *Proc. 1st ACM SIGCAS Conf. Comput. Sustain. Soc. COMPASS 2018*, 2018. DOI: 10.1145/3209811.3209815.

[125] K. Alhazmi, W. Alsumari, I. Seppo, L. Podkuiko, and M. Simon, “Effects of annotation quality on model performance,” *3rd Int. Conf. Artif. Intell. Inf. Commun. ICAIIC 2021*, pp. 63–67, 2021. DOI: 10.1109/ICAIIC51459.2021.9415271.

[126] J. Kolluri, V. K. Kotte, M. S. B. Phridviraj, and S. Razia, “Using Novel L1 / 4 Regularization Method,” *IEEE Access*, no. Icoei, pp. 934–938, 2020.

[127] C. R. Naguri, “Recognition of Dynamic Hand Gestures from 3D Motion Data using LSTM and CNN architectures,” 2017. DOI: 10.1109/ICMLA.2017.00013.

[128] J. C. Núñez, R. Cabido, J. J. Pantrigo, A. S. Montemayor, and J. F. Vélez, “Convolutional Neural Networks and Long Short-Term Memory for skeleton-based human activity and hand gesture recognition,” vol. 76, pp. 80–94, 2018. DOI: 10.1016/j.patcog.2017.10.033.

[129] F. Shaheen and B. Verma, “An ensemble of deep learning architectures for automatic feature extraction,” *2016 IEEE Symp. Ser. Comput. Intell. SSCI 2016*, 2017. DOI: 10.1109/SSCI.2016.7850047.

[130] M. R. Gunawan and E. C. Djamal, “Spatio-Temporal Approach using CNN-RNN in Hand Gesture Recognition,” in *2021 4th Int. Conf. Comput. Informatics Eng.*, 2021, pp. 385–389. DOI: 10.1109/IC2IE53219.2021.9649108.

[131] R. Xin, J. Zhang, and Y. Shao, “Complex network classification with convolutional neural network,” *Tsinghua Sci. Technol.*, vol. 25, no. 4, pp. 447–457, 2020, ISSN: 18787606. DOI: 10.26599/TST.2019.9010055. arXiv: 1802.00539.

[132] K. Gouhara, T. Watanabe, and Y. Uchikawa, “Learning Process,” *IEEE Int. Jt. Conf. neural Netw.*, pp. 746–751, 1991. DOI: doi:10.1109/ijcnn.1991.170489.

Appendix A

Data acquisition protocol

Data acquisition protocol using the Leap Motion Controller (LMC) for hand gesture recognition research:

1. Setup and Calibration:
    - Ensure that the Leap Motion Controller is securely connected to a compatible computer system and positioned on a stable surface.
    - Calibrate the device according to manufacturer instructions to ensure accurate hand tracking and gesture detection.
2. Participant Preparation:
    - Recruit participants for the study, ensuring they meet any inclusion criteria related to age, hand dominance, or previous hand injuries.
    - Provide participants with informed consent forms detailing the study objectives, procedures, and any associated risks.
3. Environment Setup:
    - Ensure the data acquisition environment is well-lit and free from obstructions to minimize interference with hand tracking.
    - Position the Leap Motion Controller at a comfortable height and angle for participants, ensuring optimal hand visibility and tracking accuracy.
4. Data Collection Procedure:
    - Instruct participants to sit comfortably in front of the Leap Motion Controller, ensuring a distance of approximately 20 cm between their hand and the sensor.

- Explain the task instructions clearly, detailing the specific hand gestures or movements required for data collection.
- Begin data recording using the Leap Motion Controller software, capturing both spatial positions and directional data of the participant's hand movements.
- Instruct participants to perform a series of predefined hand gestures or movements systematically, ensuring consistent execution across all trials.
- Record demographic information such as age, gender, and any relevant medical history for each participant to facilitate subsequent analysis.

5. Data Acquisition Parameters:
    - Set the sampling rate of the Leap Motion Controller to ensure sufficient temporal resolution for capturing hand movements accurately.
    - Determine the duration of each data recording session, balancing the need for comprehensive data collection with participant comfort and engagement.
6. Data Quality Control:
    - Monitor data acquisition in real-time to identify and address any issues such as tracking errors or signal noise that may affect data quality.
    - Verify the integrity of the recorded data, checking for completeness and consistency across all collected samples.
7. Participant Feedback and Debriefing:
    - Provide participants with an opportunity to provide feedback on their experience with the data acquisition process, addressing any concerns or questions they may have.
    - Debrief participants on the study objectives and outcomes, ensuring they understand how their data will be used and the significance of their participation.
8. Data Storage and Management:
    - Store all collected data securely in a designated database or repository, ensuring compliance with data protection regulations and ethical guidelines.
    - Implement data management protocols to organize and catalog the acquired data, including appropriate labeling and documentation for future analysis.

9. Ethical Considerations:

    - Obtain ethical approval from relevant institutional review boards or ethics committees prior to commencing data collection, ensuring adherence to ethical principles and guidelines.
    - Safeguard participant privacy and confidentiality by anonymizing personal information and obtaining informed consent for data usage and publication.

10. Data Analysis and Interpretation:

    - Conduct comprehensive data analysis using appropriate statistical and machine learning techniques to explore patterns, relationships, and trends within the acquired dataset.
    - Interpret the findings of the data analysis in relation to the research objectives, drawing meaningful insights and conclusions to contribute to the advancement of hand gesture recognition research.

By following this detailed data acquisition protocol, researchers can ensure systematic and standardized collection of high-quality data using the Leap Motion Controller for hand gesture recognition research.