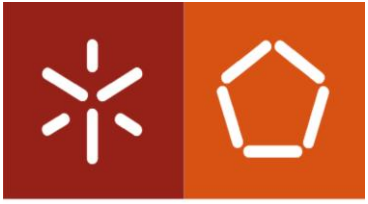


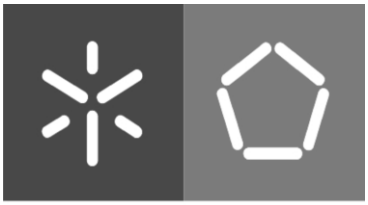
Universidade do Minho
Escola de Engenharia

Bruno Miguel Fernandes Magalhães

Leap Motion and Artificial Intelligence for surgical navigation: automatic hand gestures recognition

outubro de 2023

outubro de 2023



Universidade do Minho
Escola de Engenharia

Bruno Miguel Fernandes Magalhães

Leap Motion and Artificial Intelligence for surgical navigation: automatic hand gestures recognition

Mestrado em Engenharia Informática

Sistemas Inteligentes e Engenharia do Conhecimento

Trabalho realizado sob a orientação de

Professora Doutora Cristina P. Santos

outubro de 2023

DIREITOS DE AUTOR E CONDIÇÕES DE UTILIZAÇÃO DO TRABALHO POR TERCEIROS

AGRADECIMENTOS

O terminar desta dissertação dita também o fecho de um capítulo da minha vida que irei relembrar com saudade e ternura. Foram dois anos muito sofridos com muita dor de cabeça e algum choro pelo meio, mas que permitiram crescer como pessoa e como profissional. O processo de desenvolvimento da tese testou a minha ansiedade e nervosismo, mas permitiu alargar o meu conhecimento na área e com algum ânimo e emoção me permitirá dizer que me tornei mestre em Engenharia Informática. Estes anos seriam impossíveis de ser concretizados sem o apoio de pessoas muito importantes na minha vida que merecem agradecimento pela sua contribuição na minha vida.

Em primeiro lugar, aos meus pais que sempre trabalharam para não me falhar nada e para poder ter acesso a um curso superior que me possa dar a vida que não tiveram chance de ter. A paciência interminável para me aturar nos momentos de maior stress e cansaço providenciado um ombro amigo e a atenção necessária para continuar no caminho certo.

Em segundo lugar, à minha irmã pela atenção demonstrada durante o meu mestrado e pela ajuda a lidar com os problemas que surgiram no correr destes últimos dois anos.

Em terceiro lugar, aos meus amigos Rodrigo, Mendes, Leandro, Barros, Catarina, Renata, Beatriz, Diogo, Duarte e Pedro que prezo pelas saídas ocasionais, fossem elas para abrandar o ritmo e desligar um pouco dos problemas, fossem para dar conselhos e incutir juízo na minha cabeça.

Ao Roberto e à Diana, que desde o início foram muito prestáveis e incansáveis. Os seus valiosos conselhos em momentos de indecisão foram vitais para a realização deste documento. Por fim, agradecer à professora Doutora Cristina P. Santos pela oportunidade de trabalhar neste projeto, pela sua disponibilidade e conselhos que permitiram concluir este último ano.

Um muito obrigado a todos!

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

Resumo

Globalmente, 310 milhões de cirurgias são realizadas a cada ano e existe uma probabilidade de 2% a 5% de infeções do local cirúrgico. Problemas relacionados com a esterilização de equipamentos são uma das razões. Estas infeções impactam negativamente a saúde física e mental do paciente comprometendo a sua qualidade de vida. As cirurgias assistidas por computador estão a ajudar os cirurgiões a realizar operações mais seguras e a permitir aos pacientes menos tempo de recuperação. No entanto, este meio de interação geralmente depende de dispositivos de contato físico, como rato e teclado, que expõem a sala de cirúrgica a condições assépticas. O *Leap Motion* ultrapassa o problema dos dispositivos físicos uma vez que não precisa de nenhum tipo de interação física.

Esta dissertação tem como objetivo conceber, desenvolver e validar uma abordagem de interação homem-computador intuitiva e sem contacto, baseada no reconhecimento automático de gestos manuais através do *Leap Motion*, seguindo uma conceção centrado no utilizador. Para tal, foi primeiramente realizado um protocolo junto dos utilizadores finais para determinar que gestos eram mais intuitivos. Posteriormente, foram criados dois grandes *datasets* (um de imagens da mão e um com características da mão) para alimentar modelos de inteligência artificial que pudessem reconhecer os gestos manuais de qualquer pessoa. O melhor modelo desenvolvido, com 96.25% de precisão nos dados de teste, foi baseado no algoritmo *Support Vector Machine* e foi, de seguida, integrado na ferramenta de reconhecimento de gestos manuais que através das previsões do modelo, executa a respetiva ação no ecrã, removendo a necessidade de periféricos com contacto físico.

A partir de uma validação preliminar realizada junto de voluntários da Universidade do Minho e uma validação clínica realizada junto de cirurgiões do hospital Trofa Saúde Braga Centro, verificou-se que os utilizadores demoram mais tempo a realizar o mesmo conjunto de tarefas com a ferramenta de deteção de gestos manuais do que com o uso tradicional do rato. Contudo, foi possível observar que há uma curva de aprendizagem da ferramenta e que estes tempos diminuem com a experiência. Por fim, o *System Usability Scale*, que é um teste padronizado de avaliação de usabilidade, revelou que a aplicação desenvolvida atinge um resultado de 76.67 ± 9.86, porém há uma perceção de usabilidade maior na validação preliminar do que na validação clínica (67.5 ± 6.37). Através de uma última questão aberta pôde-se ainda perceber que a sensibilidade do cursor é o que precisa de mais atenção e constitui o ponto principal do trabalho futuro, juntamente com melhorias na interface gráfica.

Palavras-Chave: Cirurgia assistida por computador, Inteligência Artificial, Interação Humano-Computador, *Leap Motion*, Reconhecimento de gestos manuais

ABSTRACT

Globally, 310 million major surgeries are performed each year, and there is a 2% to 5% chance of surgical site infections. Problems related to equipment sterilization are one of the reasons. These infections negatively affect the patient's physical and mental health, comprising their quality of life. Computer-assisted surgeries are helping surgeons perform safer surgical interventions and allowing patients to have shorter recovery times. However, this means of interaction usually relies on physical contact devices, such as a mouse and a keyboard, which exposes the operating room to aseptic conditions. The leap motion overcomes the problem of physical devices as it does not require any kind of physical interaction.

This dissertation aims to design, develop, and validate an intuitive, touch-free human-computer interaction approach based on the automatic recognition of hand gestures through Leap Motion, following an end-user-centred design. To achieve this, a protocol was carried out with end-users to determine which gestures were most intuitive. Subsequently, two large datasets were created (one with hand images and one with hand features) to feed artificial intelligence models that could later recognise anyone's hand gestures. The best model developed, with 96.25% accuracy on the test data, was based on the Support Vector Machine algorithm and was then integrated into the hand gesture recognition tool, which using the model's predictions, performs the corresponding action on the screen, removing the need for peripherals with physical contact.

Preliminary validation with volunteers from the University of Minho and clinical validation with surgeons from the *Trofa Saúde Braga Centro* hospital showed that users take longer to perform the same set of tasks with the hand gesture recognition tool than with the traditional mouse control. However, it was possible to observe that there is a learning curve for the tool and that these times decrease with experience. Finally, the System Usability Scale, which is a standardised usability evaluation test, revealed that the application developed achieves a score of 76.67 ± 9.86, but there is a greater perception of usability in the preliminary validation than in the clinical validation (67.5 ± 6.37). A final open question also revealed that the cursor sensitivity is what needs more attention and is the focus of future work, along with improvements to the graphic interface.

KEYWORDS: Artificial Intelligence, Computer-assisted surgery, Hand Gestures Recognition, Human-Computer Interaction, Leap Motion

CONTENTS

List of Figures

LIST OF TABLES

List of Acronyms

AI – Artificial Intelligence

API – Application Programming Interface

BiRDLab - Biomedical Robotic Devices Laboratory

CAS - Computer-Assisted Surgery

CMEMS - Centre of MicroElectroMechanical Systems

CNN – Convolutional Neural Network

DL – Deep Learning

DICOM – Digital Imaging and Communication in Medicine

ELM – Extreme Learning Machine

GPU - Graphics Processing Unit

GUI - Graphical User Interface

HCI – Human-Computer Interaction

KPI – Key Performance Indicator

LM – Leap Motion

LSTM – Long Short-Term Memory

ML - Machine Learning

MLP – Multi-Layer Perceptron

NN – Neural Network

PCA - Principal Component Analysis

PRISMA – Preferred Reporting Items for Systematic Reviews and Meta-Analyses

ReLU - Rectified Linear Unit

RQ – Research Question

SLR – Sign Language Recognition

SUS – System Usability Scale

SVM – Support Vector Machine

1. INTRODUCTION

This manuscript presents the work developed in the scope of the fifth year of the Master's in Informatics Engineering, at the University of Minho, during the academic year of 2022-2023.

The academic year was passed working in the Biomedical Robotic Devices Laboratory (BiRDLab), included in the Centre of MicroElectroMechanical Systems (CMEMS), at the University of Minho, Guimarães, Portugal. During this period, an automatic hand gesture recognition solution for surgical navigation was developed and validated. This system aims to improve operating room procedures by giving surgeons a more sterile and intuitive way to control computer devices. All the methods, results, and conclusions are detailed in this document.

1.1 Motivation and Problem Statement

Globally, 310 million major surgeries are performed each year [1] and with an ever-increasing world population, the trend of this number is to increase. During a medical procedure, there is a 2% to 5% chance of surgical site infections [2] due to various reasons being one of them problems related to equipment sterilization. Despite the low percentage, in a world where 310 million surgeries are performed annually, it means that between 6 200 000 and 15 500 000 people still suffer an unwanted infection.

This complication can negatively affect the patient's physical and mental health. The infection can cause pain, susceptibility to other complications, delayed healing, or even the need for a second surgery [3]. Prolonged hospitalization, as well as the consequent time away from work, comprises the patient's quality of life, and professional and social inclusion, in addition to leading to a decrease in monetary income, which may also affect the lives of dependent family members [3]. Thus, it is important to reduce the number of avoidable infections.

Technology has slowly integrated into almost every aspect of human life due to the immense potential it has to increase the overall quality of life. This presence is also felt in the medical field, where new developments try to improve patients' safety and the doctor's quality of work. Computer-assisted surgery (CAS) is one such example, an emerging technology where the surgeon takes advantage of technology for surgical planning and guidance, using digital images usually displayed through a computer screen [4]. Their demand and interest are also described by the global market, expected to grow from 6.1 billion dollars in 2020 to 11.6 billion dollars in 2025 [5]. CAS is a technology increasingly implemented and used in numerous areas of medicine, as it brings immense advantages to surgery such

as improving the precision of certain surgical navigations leading to better patient outcomes. In the dental field [6], for example, the implant position in fully edentulous patients achieved greater accuracy when using CAS than when performed freehand, limiting the chance of human error. Also, for orthognathic surgery [7], there is evidence to support the use of CAS, based on less time used in the preparatory stages of surgery, more accurate surgical planning, and better surgical outcomes. The increasing number of peripheral devices and equipment in the operating room has contributed negatively to surgical site infections and it represents a real concern of surgical navigation solutions [8].

The CAS systems comprise human-computer interaction (HCI) solutions that typically rely on the use of a mouse and keyboard [9]. In the context of an operating room, several devices have already been tested to give surgeons a more natural and usable means of interaction, such as gloves, trackballs, and joysticks [9]. Although these devices are practical and easy to manoeuvre, they require physical contact, risking exposure to aseptic conditions and loss of time [10]. Therefore, to solve this problem, some methods have been developed, such as hand gesture recognition [11] and speech recognition [12], which use deep learning (DL) models to learn patterns in a dataset and label data. The noise in the operating room can affect the proper functioning of a speech recognition solution, whereas hand gesture recognition by a camera is unaffected by environmental conditions.

Leap Motion (LM), being an optic device, overcomes the problem of physical devices as it does not require any kind of physical interaction [13], avoiding the need to sterilize the equipment between each surgery and saving time that can be used for more surgeries, while it is more natural and practical for the surgeon performing the CAS [14]. Current scientific works already make use of this technology, as it can be used to recognise hand gestures to manipulate medical images through a touchless graphical user interface (GUI) [15], and can be used in operating rooms to reduce the risk of contamination during medical procedures [16]. However, there is a limited level of effort within the medical field to solve this problem. Further, most of the studies do not demonstrate to be end-user centred [16]–[18], which means that the selected gestures may not necessarily be the easiest and most intuitive for the final users to operate the solution developed. Thus, there is a need to fill these gaps in the literature by contributing with an automatic hand gesture recognition tool developed from an end-user-centred gesture dataset, as well as new models with better performances. This work will be integrated into the NavPI surgical navigation system, already developed in BiRDLab.

1.2 Goals and Research Questions

The ultimate goal of this dissertation is to design, develop, and validate a LM-based application for hand gesture recognition, to be integrated with the NavPI surgical navigation setup. Artificial Intelligence (AI)-based algorithms for automatic hand gesture recognition will be developed using images and features of the hand, provided by LM. Validation experiments and data acquisition will be conducted in a laboratory setting, for preliminary validations, and at the *Trofa Saúde Braga Centro* Hospital, performed by the surgeons (end-users), for final validations. Expected results include an automatic and specialized hand gesture recognition to be used in surgical navigation, to improve the quality of the surgeon's interaction with the computer, as well as the surrounding environment. Quantitative results from the experimental validation must prove the effectiveness of the developed tool by assessing the time and precision differences in comparison with the use of mouse, as the traditional gold standard method. Qualitative results must demonstrate an easier, more intuitive, and ergonomic solution to control the surgical navigation software.

To reach this main goal, it is necessary to achieve the following step objectives, measurable and verifiable through associated Key Performance Indicators (KPI):

- **Objective 1:** Literature review of the existing AI-based solutions for hand gesture recognition using LM, as well as their current limitations. This is addressed in Chapter 3.
- **Objective 2:** Define the system's specifications and requirements, following end-users-centred approaches focusing on usability. This is addressed in Chapter 3 and Chapter 5. KPI: at least 3 surgeons (end-users).
- **Objective 3:** Create two datasets: one with hand images, and another with hand features regarding its position and rotation, provided by the LM Application Programming Interface (API). This is addressed in Chapter 6. KPI: at least 15 subjects in the dataset and more than 200 000 samples.
- **Objective 4:** Design and develop AI-based algorithms for automatic hand gesture recognition. This is addressed in Chapter 6. KPI: accuracy above 90% on the test data and inference time below 0.1 seconds.
- **Objective 5:** Development of an application that integrates the algorithms developed, to control the computer mouse. This is addressed in Chapter 6.
- **Objective 6:** Validate the developed LM application with new subjects, including the end-users. The validation protocol should be designed and implemented to evaluate the tool's operability and effectiveness. The tool's acceptability, usability, and user experience should be accessed

through a usability questionnaire and user feedback. This is addressed in Chapter 7. KPI: software usability above average according to the System Usability Scale (SUS) [19], resorting to surgeons.

The outlined objectives will allow answering the following Research Questions (RQs):

- **RQ1**: What specifications should be considered for the development of a LM hand gesture recognition application fitted for surgical navigation? The answer is included in Chapter 2.
- **RQ2**: What are the most appropriate gestures needed to control the LM application in a surgical environment? The answer is included in Chapter 5.
- **RQ3**: Can hand gesture recognition achieve good performance using AI algorithms for real-time use? The answer is included in Chapter 6.
- **RQ4**: Can a gesture recognition-based solution be intuitive for controlling a surgical navigation application? The answer is included in Chapter 7.

1.3 Contribution to Knowledge

The main contribution of this dissertation to knowledge are:

- A review on DL based hand gesture recognition tools using LM;
- Study with clinicians to define the most intuitive mouse control gestures based on end-users;
- A novel hand gesture recognition tool for mouse control in surgical navigation, following an end-user-centred design;
- Validation of the developed tool with the final users to motivate further research in the topic.

1.4 Manuscript Outline

This manuscript is organized into 8 chapters, as follows.

Chapter 2 gives a theoretical introduction to AI, Machine Learning (ML), and DL. It also gives a brief overview of the architectures available in each of these areas.

Chapter 3 presents the state-of-the-art of current LM solutions for hand gesture recognition. The specifications of the studies are presented and discussed, regarding the study focus, dataset type and size, DL architectures, evaluation metrics, and solution validation.

Chapter 4 describes the proposed methodology to achieve the hand gesture recognition tool, the hardware and software required to develop the tools to accomplish all the objectives, as well as the definition of the actions required to replace the computer mouse.

Chapter 5 describes the protocol used in the early stages of the work to understand, with the surgeons, the most intuitive hand gestures for replicating the actions offered by the computer mouse.

Chapter 6 outlines the protocol used to create the dataset, the data analysis and processing of the dataset, the architectures explored for the two datasets, the results of the models developed, and the development of the hand gesture recognition tool for mouse control.

Chapter 7 comprises the validation protocol used to assess the quality of the hand gesture recognition tool.

Chapter 8 addresses the conclusions of this dissertation, answering the RQs and appointing future work.

2. THEORETICAL INTRODUCTION

AI can be traced back to 1950, when Alan Turing, often referred to as the father of computer science, famously pondered if machines could think and introduced the Turing Test to determine if a computer could demonstrate the same intelligence as a human [20]. The definition of AI has changed over the years, but it generally consists of developing algorithms that allow computers to analyse data and learn over time. This field also encompasses the subfields of ML and DL, which are usually used interchangeably but mean different things. It is therefore important to show the difference between them.

2.1 Machine Learning

Arthur Samuel is credited with coining the term ML in his work on the game of checkers. This branch of AI focuses on using data and algorithms to mimic the way humans learn [21]. ML is a term for solving problems by helping machines discover their own algorithms, rather than humans developing prohibitive high-cost algorithms to tell the machine what to do [22]. This field includes four different learning approaches, each proposing a different way to solve a problem: supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning [21]. In Figure 2.1, it is possible to see an outline of the ML approaches.

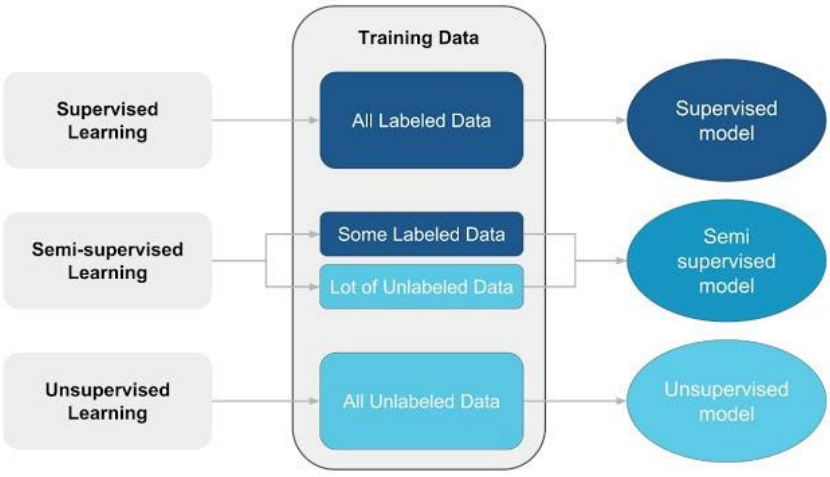


Figure 2.1 – Outline of the ML approaches [23].

In the supervised learning approach, labelled datasets are used to train algorithms to classify outcomes. The input data is fed into the model, and through the labelled classes, the model adjusts its weights until the data is properly fitted. Neural networks (NNs), linear and logistic regression, and random forest are some of the methods used in supervised learning. NNs are inspired by the workings of the human brain, copying the notion of a neuron and the communication between them. NNs consists of an

input layer, one or more hidden layers and an output layer for prediction. These layers are made up of several nodes, which are the analogue term for the neuron, and each carries a weight and a threshold that decides whether a given neuron is activated and propagates the information to another neuron. This architecture learns by looking at the same data several times and adjusting the weights each time to minimize a cost function to the point of convergence [24]. In Figure 2.2, it is possible to see an example of a NN architecture.

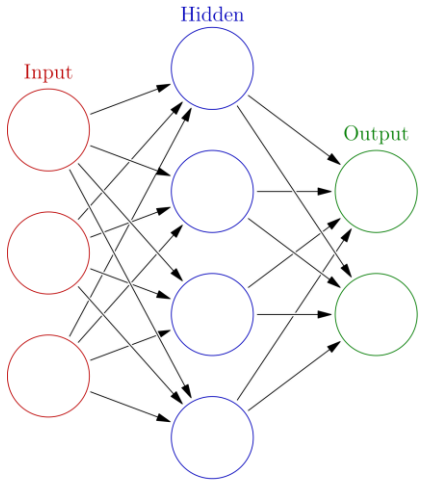


Figure 2.2 – Example of a NN architecture [25].

Linear regression and logistic regression are both used to estimate the relationship between a dependent variable (label or target to be predicted) and one or more independent variables (features). The main difference is that linear regression is used to predict a continuous variable and logistic regression operates to predict categorical variables. Linear regression finds a line that best fits the data using a calculation method that minimizes the discrepancies between predicted and output values [26]. The logistic regression analyses the possible outcomes and calculates a distribution of probabilities between 0 and 1 to classify the chance of a given event occurring [27]. In Figure 2.3, it is possible to see an example of each approach and the difference between them.

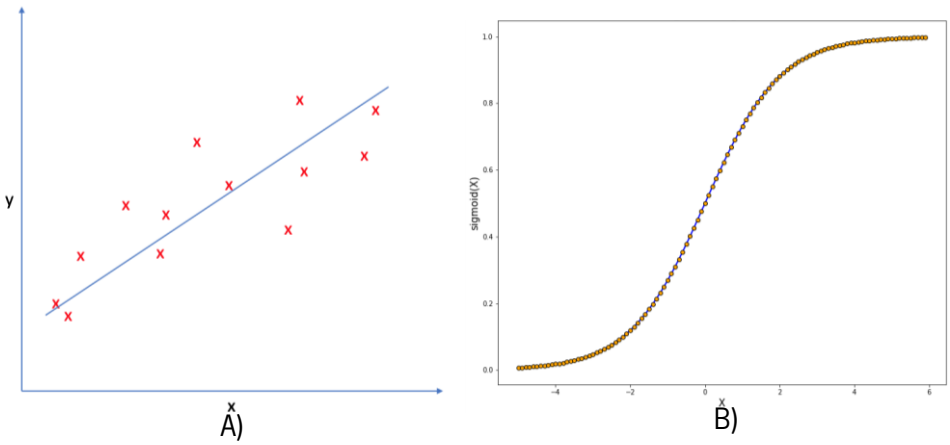


Figure 2.3 – Example of A) linear regression [28] and B) logistic regression [29].

Random Forest is an algorithm that combines the output of several decision trees to produce a single result. The decision tree structure is comprised of a root node, decision nodes, and leaf nodes, forming a hierarchical, tree-like structure. The root node is at the beginning of a decision tree, where the dataset starts to be divided based on the various features. The decision nodes result from the splitting of root nodes, and they represent decisions within the tree. The leaf nodes are nodes where further splitting is not possible and indicate a final classification. The Figure 2.4 shows a brief overview of the Random Forest algorithm.

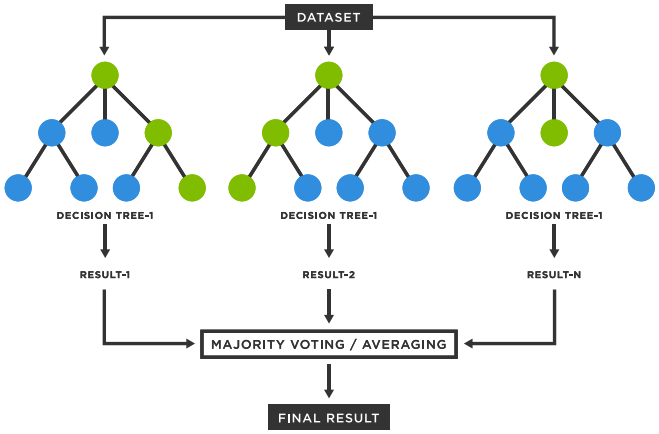


Figure 2.4 – Overview of the Random Forest algorithm [30].

In the unsupervised approach, the algorithms analyse unlabelled datasets to discover hidden patterns and cluster the data into groups. This method of learning is ideal for data analysis, customer segmentation and to reduce the number of features in a model. Principal Component Analysis (PCA) and K-Means clustering are good examples of algorithms used in this approach. PCA is used to reduce the dimensionality of a dataset by finding new variables that are linear functions of the variables in the original dataset. These new variables maximise variance and are uncorrelated with each other [31]. The Figure 2.5 shows an application of the PCA algorithm.

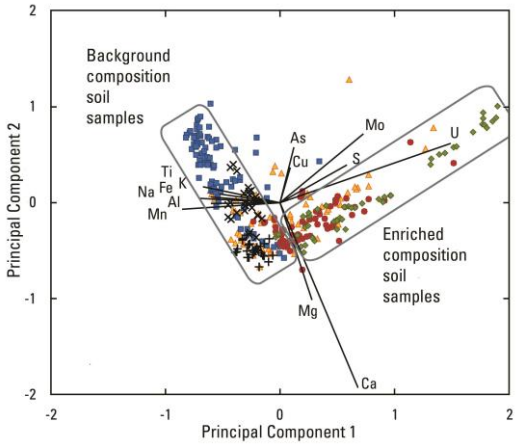


Figure 2.5 – Application of the PCA algorithm [32].

K-Means is an algorithm that assigns a set number of data points into a given number of clusters (K) to group similar data points together. It does this by assigning each data point to the cluster with the nearest centroid. It then recalculates the centre of the cluster by taking the average of its data points. This process is repeated iteratively until convergence is reached [33]. In Figure 2.6, it is possible to see an example with three clusters and their centroids represented by the red triangle.

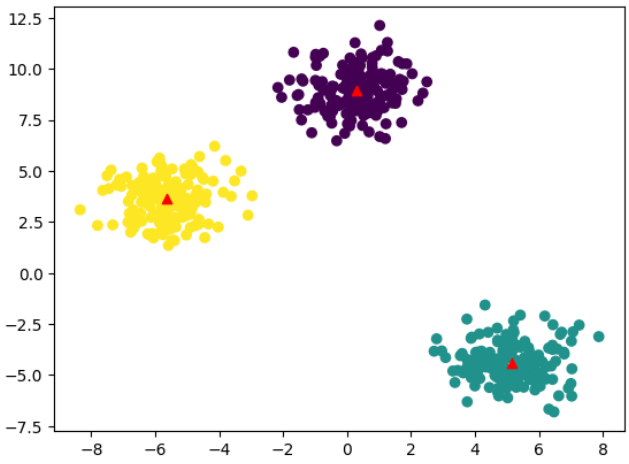


Figure 2.6 – Example of a K-Means application [34].

Semi-supervised learning approach is a balance between supervised and unsupervised learning. It trains a model using a small, labelled dataset, which is later used to classify and label the remaining dataset. This can be useful when labelling a very large dataset is too expensive.

Reinforcement learning approach is similar to supervised learning in the sense that it receives feedback, but instead of the feedback coming from labelled data, it comes from states and actions. The algorithm has a given set of actions and an end goal, and it learns by being rewarded (reinforcement signal) when it reaches the end goal [35].

2.2 Deep Learning

DL is a subfield of ML that, as the name suggests, essentially enables deeper learning by stacking more layers in a NN. Similar to ML, it uses data and algorithms to mimic the way humans learn, but thanks to the recent developments in Graphics Processing Unit (GPU) acceleration, the capabilities of deeper architectures are more accessible. As DL is a subfield of ML, they are not very different, but they do differ on one key aspect. While ML relies on hand-crafted features, DL automates most of the feature extraction process, allowing humans to simply feed the algorithms with a large amount of data and let them figure out which pieces of information are more important [36]. In DL, the supervised and

unsupervised learning approaches also exist with the same concepts, but with different and deeper methods of learning.

In the supervised learning, which uses labelled datasets, some examples of architectures are the Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and Gated Recurrent Unit [37]. The structure of the CNN was inspired by neurons in human brain. Unlike conventional fully connected networks, CNN's shared weights and local connections make full use of two-dimensional input data structures, making it ideal for image signals. These characteristics make CNN a great architecture for computer vision, such as image classification, speech processing, and face recognition [38]. In Figure 2.7, it is possible to see an example of a CNN used for image classification.

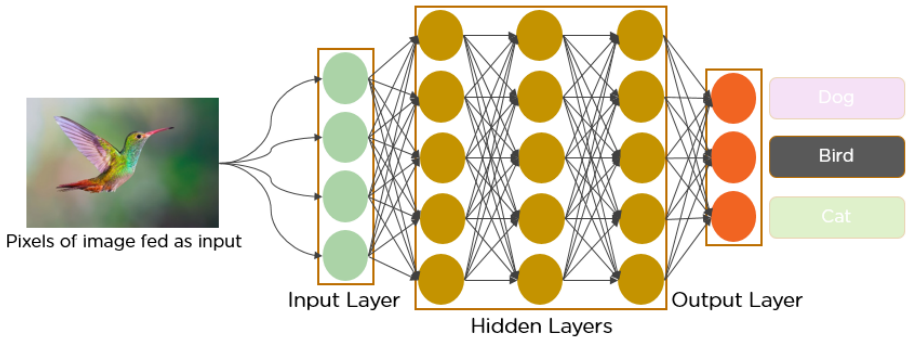


Figure 2.7 – Example of a CNN used for image classification [39].

LSTM is a Recurrent NN architecture that introduces the concept of a memory cell that can hold its value for a variable amount of time as a function of its inputs, allowing the cell to remember. It achieves this by having three gates that control the flow of information. The input gate that controls when new information can enter the memory, the forget gate that controls when existing information is forgotten, and the output gate that controls when the information in the cell is used in the output. These characteristics help to achieve a higher precision in the modelling of time-variant behaviour [40] and make this architecture ideal for time-series problems. Gated Recurrent Unit is a simplified version of the LSTM architecture that replaces the concept of the memory cell with a hidden state using only a reset gate and an update gate. The reset gate controls whether to ignore the previous hidden state and reset it with the current input it is receiving, allowing the cell to drop information that is found irrelevant in a later state. The update gate controls how much information is passed from the previous state to the current state. This change makes the architecture easier to compute and implement, resulting in more efficient train times [41]. However, the more complex structure of the LSTM can lead to better results with more data. The nature of this architecture also makes it a suitable candidate for solving time-series problems. In Figure 2.8, it is possible to see the architectures from the LSTM and Gated Recurrent Unit and analyse their differences.

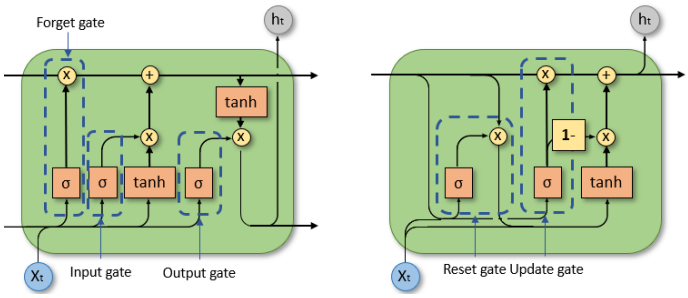


Figure 2.8 – Overview of the LSTM and Gated Recurrent Unit architectures [42].

In the unsupervised learning approach, there is no labelled data and no target label to predict. The goal is to learn representations of the data without supervision to capture underlying semantic information that can be transferable to tasks such as visual recognition and segmentation [43]. The architectures available in this spectrum are Self-Organizing Maps and AutoEncoders. Self-Organizing Maps is an artificial NN algorithm that starts with a fixed number of nodes. Random data inputs are presented to each node individually and each node calculates the distance to the input, winning the node that is closest. Over iterations, these distances begin to stabilise, and it is possible to see centroids throughout the data, clustering the data into classes [44]. The Figure 2.9 shows an example result of a Self-Organizing Map.

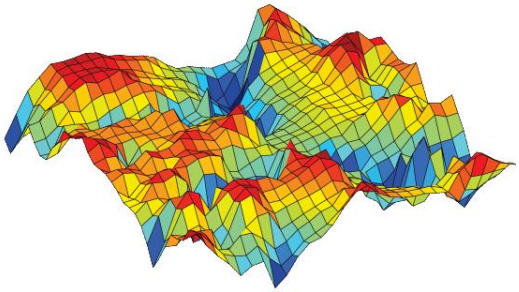


Figure 2.9 – Visual representation of an example Self-Organizing Map result [45].

Autoencoders consist of three layers and two components. The encoder compresses the input into a meaningful representation of the data in the second layer, and the decoder reconstructs the data in a way that is as similar as possible to the original input. In Figure 2.10, it is possible to see an overview of the architecture of an Autoencoder. The main purpose of this architecture is to learn an informative and generalized representation of the data in an unsupervised manner [46].

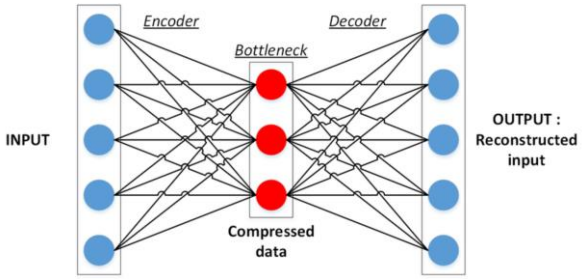


Figure 2.10 – Overview of the architecture of an Autoencoder [47].

3. REVIEW ON DEEP LEARNING-BASED HAND GESTURE RECOGNITION USING LEAP MOTION

A review of DL-based hand gesture recognition tools is essential to accelerate future research, considering the benefits that these technologies can bring to the operating room and surgical navigation procedures. Current reviews on this topic, like the ones from SHI et al. [48] and Hirafuji Neiva et al. [49], focus on the techniques used for classification and the accuracies obtained. However, they lack the use of a user-centred approach and do not consider the real-world scenarios that may change the specifications considered. Thus, this chapter reviews the approaches taken on hand gesture recognition with LM device, advancing current literature reviews and aiming to answer the following RQ: What specifications should be considered for the development of a LM hand gesture recognition application fitted for surgical navigation?

3.1 Methodology

The studies included in this review were searched for in October 2022 on Scopus (search field: "Article title, Abstract, Keywords"), IEEE (search field: "All Fields"), Web of Science (search field: "All Fields"), and PubMed (search field: "All Metadata") databases using the keywords "leap motion", "deep learning" and "artificial intelligence". The following combination of such terms was used: "leap motion" AND ("deep learning" OR "artificial intelligence"). All published studies were considered.

The reference list of all the relevant studies was checked. Among the resulting studies, only those comprising all the eligibility criteria were included in this review. The inclusion criteria were: (1) use LM; (2) use DL techniques; (3) work on hand gesture recognition. The exclusion criteria were: (1) do not use LM; (2) only uses ML techniques; (3) does not use or describe models; (4) does not have open access; (5) uses extra resources for data fusion; (6) is written in a non-international language; (7) it does not operate on learning hand gestures; (8) does not consist of original research. The following specifications were extracted: study focus, dataset type and size, DL architecture, the evaluation metrics of the corresponding models, and the solution validation.

3.2 Results

This review followed the search strategy based on Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) (Figure 3.1). From the four databases used, 358 records were identified, resulting in 326 studies after the duplicates' removal. Then, 86 studies were removed after the title and abstract reading, and 211 studies were excluded during the full-text reading for containing one or more exclusion criteria. A total of 29 studies were included in this review.

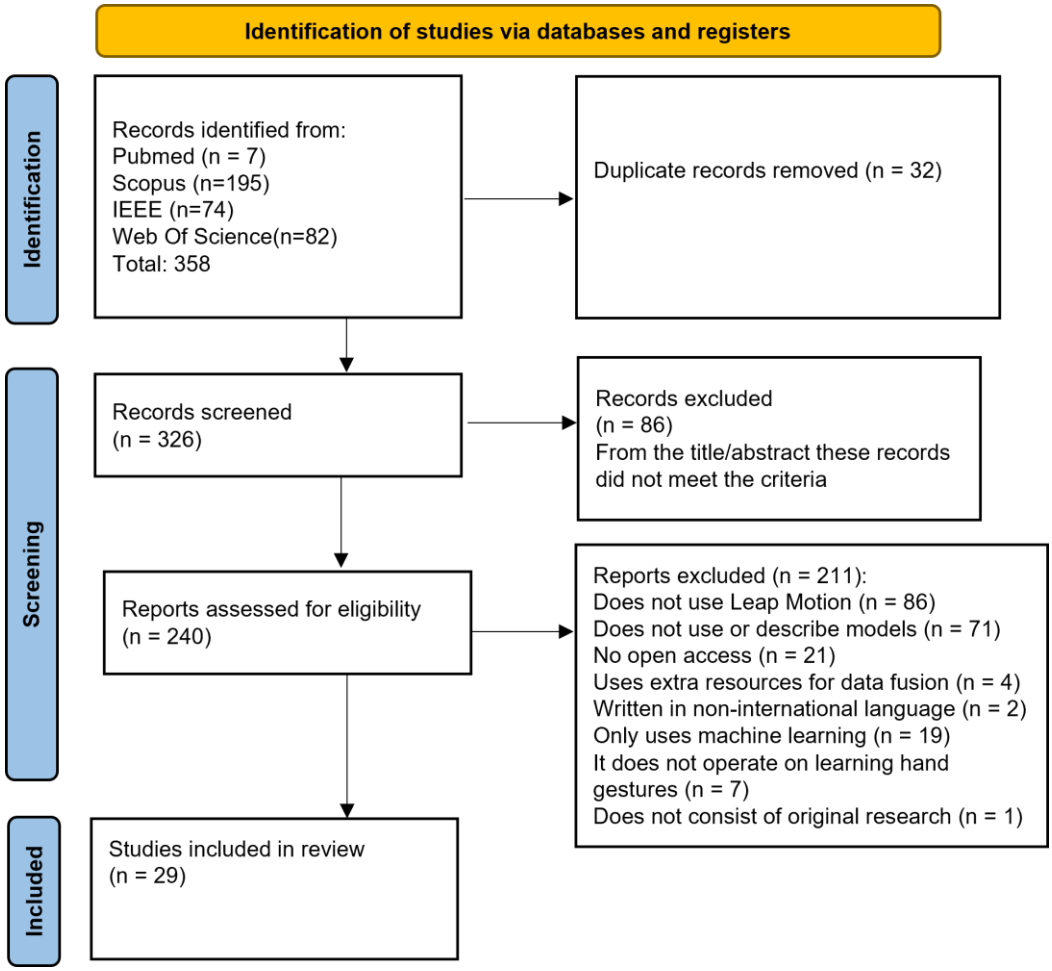


Figure 3.1 – Flow diagram of search strategy based on PRISMA.

The extracted characteristics of the included studies are shown in Table 3.1.

Table 3.1 – Characteristics of the included studies

Study	Study Focus	Dataset Type	Dataset Size	DL Architectures	Metrics of Evaluation	Solution Validation
Abdullahi et al. (2022) [50]	Sign Language Recognition (SLR)	Feature-based	4000 samples	Multi-stack deep Bidirectional Long Short-Term Memory (LSTM)	Accuracy, Fowlkes-Mallows Index, Matthew Correlation Coefficient, Sensitivity, Specificity, Bookmaker Informedness, Jaccard Similarity Index	Not mentioned
Abdullahi et al. (2022) [51]	SLR	Feature-based	5700 samples	Fast Fisher Vector Bidirectional LSTM	Accuracy, Precision, Recall, F1-Score	Not mentioned
Ameur et al. (2020) [52]	Entertainment	Feature-based for the two datasets	6600 samples | 9600 samples	Basic Unidirectional LSTM, Deep LSTM, Bidirectional LSTM, Hybrid Bidirectional Unidirectional LSTM	Accuracy, Execution time	Not mentioned
Ameur et al. (2020) [15]	Medical	Feature-based for the two datasets	6600 samples | 9600 samples	Multi-Layer Perceptron (MLP)	Accuracy	A simple graphical user interface (GUI) to manipulate DICOM images using the Leap Motion (LM) Camera and the model developed
Brock et al. (2020) [53]	Entertainment	Feature-based	1472 samples	Convolutional Neural Network (CNN)	Accuracy, Precision, Recall, F1-Score, Run time	Ten volunteers are asked to try the developed solutions and answer a survey to evaluate user impression
Caputo et al. (2020) [54]	Scientific	Feature-based	468 samples	CNN (Residual Network 50)	Accuracy, F1-Score, False Positive	Not mentioned
Enikeev et al. (2021) [55]	SLR	Image-based	800 samples	CNN	Accuracy	Not mentioned
Hu et al. (2018) [56]	Robotics	Feature-based	11062 samples	Deep Neural Network (NN)	Accuracy	Real-time drone control with the developed hand gesture recognition solution in two places on the University Campus
Hu et al. (2020) [57]	Robotics	Feature-based	11061 samples	Deep NN	Accuracy	Real-time drone control with the developed hand gesture recognition solution in two places on the University Campus
Ikram et al. (2021) [58]	Scientific	Feature-based	800 samples	CNN, CNN-Support Vector Machine (SVM)	Accuracy	Not mentioned
Katılmış et al. (2021) [59]	SLR	Feature-based	8000 samples	Extreme Learning Machine (ELM), Kernel-based ELM, MLP-ELM, Multi-Layer ELM, Multi-Layer Kernel-Based ELM	Accuracy, Run time	Not mentioned
Kritsis et al. (2019) [60]	Scientific	Feature-based	1019 samples	CNN-LSTM, Deep CNN	Accuracy, Run time	Not mentioned
Lee et al. (2021) [61]	SLR	Feature-based	2600 samples	LSTM-Recurrent NN with k-Nearest Neighbours	Accuracy, Sensitivity, Specificity	Not mentioned
Lee et al. (2020) [16]	Medical	Image-based	1000 samples	Capsule NN, CNN, Visual Geometry Group 16	Accuracy	Not mentioned
Li et al. (2022) [17]	Entertainment	Feature-based	+ than 200000 samples	Deep NN	Accuracy	Not mentioned
Li et al. (2019) [62]	Scientific	Feature-based	400 samples	LSTM-Recurrent NN, Incremental Learning	Accuracy, Loss Value, Training time	Not mentioned
Lin et al. (2020) [63]	Security	Feature-based	100 samples	NN	Accuracy	Invited 10 volunteers to leave a 2D signature on a paper and 10 3D signatures through the proposed solution
Liu et al. (2018) [64]	Robotics	Feature-based	8520 samples	MLP	Accuracy, Loss value	Not mentioned

Study	Study Focus	Dataset Type	Dataset Size	DL Architectures	Metrics Of Evaluation	Solution Validation
Mittal et al. (2019) [65]	SLR	Feature-based	942 samples	CNN followed by LSTM	Accuracy, Error Rate	Not mentioned
Oktay et al. (2020) [66]	Medical	Feature-based	16000 samples	CNN followed by LSTM	Accuracy, Recall, Precision, F1-Score, Area Under Curve, Receiver Operating Characteristics	Not mentioned
Tripathy et al. (2019) [18]	Entertainment	Image-based	12000 samples	CNN	Accuracy	Application has been tried and assessed by ten subjects that had no prior experience on how to use the application
Tyutyunnik et al. (2021) [67]	Security	Feature-based	Not mentioned	CNN	Accuracy	Not mentioned
Wang et al. (2021) [68]	SLR	Feature-based	10000 samples	Spatial-Temporal Graph Convolutional Network	Word Error Rate	Not mentioned
Wu et al. (2021) [69]	Robotics	Feature-based	800 samples	Back Propagation NN	Accuracy	Not mentioned
Yamamoto et al. (2018) [70]	Security	Feature-based	450 samples	CNN	Accuracy, False Rejection Rate, False Acceptance Rate	Not mentioned
Yamamoto et al. (2018) [71]	Security	Feature-based	450 samples	CNN	Accuracy	Not mentioned
Yang et al. (2015) [72]	Entertainment	Feature-based	Not mentioned	RNN	Recognition Rate	Not mentioned
Yasir et al. (2017) [73]	SLR	Feature-based	Not mentioned	CNN	Error rate	Not mentioned
Zhang et al. (2017) [74]	Medical	Feature-based	6826 samples	Back Propagation NN	Accuracy, Loss Value	Not mentioned

3.2.1 Study focus

The focus of the reviewed studies is very broad, however, Sign Language Recognition (SLR) is a topic that stands out from the rest. There are eight studies whose main objective is to make improvements in the recognition of several sign languages to give hearing-impaired people the ability to better communicate with non-impaired persons. Out of these eight, three focused on American SLR [50], [51], [61], one on Russian finger spelling [55] which is a subset of SLR, one on Turkish SLR [59], and one looked to improve on Bangla Sign Recognition [73]. Two studies [65], [68] did not specify the sign language they sought to improve.

Entertainment is the focus of five of the reviewed studies. Ameur et al. [52], believing that digital entertainment is a promising field, developed an approach for dynamic hand gesture recognition. Brock et al. [53] explored a game interaction with a social robot capable of playing Rock-Paper-Scissors in real-time against a human person with hand gesture inputs. Li et al. [17], in the field of experimental teaching, developed a virtual reality experience, simulating an electrician laboratory and a set of experiments, controlled through hand gestures with a LM device. Tripathy et al. [18] presented a system capable of controlling 3D objects in a 3D environment created by the Visualization Toolkit. Yang et al. [72] developed an interface system through which a user can operate a computer with hand and finger movements.

The medical field is the focus of four of the reviewed studies. Ameur et al. [15] seek to control medical images during surgical procedures via hand gestures, without contact with peripherals, to reduce surgical time and contamination risk. Lee et al. [16], with the same objective, developed a contactless gesture system to manipulate several types of computer-aided devices. Zhang et al. [74], aiming to improve Traditional Chinese Medicine, more specifically Acupuncture, combined LM with Oculus virtual headset to create a virtual training system that provides students with an easier way to learn the craft. Oktay et al. [66] proposed a new method to help differentiate between Parkinsonian Tremor and Essential Tremor, using both the postural and resting positions of the patients' hands.

Robotics is the focus of four studies. Hu et al. [56], [57] presented a hand gesture recognition system designed to control unmanned aerial vehicle flights. Liu et al. [64] took a more industrial-focused approach and combined a multi-modal interface, including voice recognition, hand motion recognition, and body posture recognition to support the emerging needs of human-robot collaboration. Wu et al. [69] used transfer learning techniques to create a solution in the gesture recognition field in which the previously learned knowledge from one robot could be transferred to another robot in a different domain facilitating the process of learning on the new robot.

The security field is also a current area of focus in the reviewed studies, with four studies. Lin et al. [63] developed a system that recognizes signatures in a 3D environment to authenticate the identity of the real signer. The work done in [67], [70] and [71] has the same goal, presenting an access control and management system that identifies the user through a numerical sequence written in the air.

The remaining five studies present a more scientific approach. The work done by Ikram et al. [58] sought to improve dynamic hand gesture recognition by using CNN with Error Break Propagation Algorithm to reduce error. Kritsis et al. [60] and Li et al. [62] look to improve the field by experimenting with different models to see which gets the most performance. Caputo et al. [54] presented a novel benchmark aimed at evaluating online gesture detection and recognition.

3.2.2 Dataset type and size

The datasets used in the reviewed studies provide information in the form of data points to the models to help them solve a wide variety of AI challenges. These datasets have a varying range of sizes.

By analysing Table 3.1, it is possible to visualize two types of datasets (i.e., feature-based, and image-based) that were used to feed the model. Twenty-six studies used hand feature-based datasets [15], [17], [50]–[54], [56]–[74] and three studies used hand image-based datasets [16], [18], [55].

The size of the datasets ranges from 100 samples to 200000 samples. In the ranges from 100 to 1000, 1001 to 10000, and 10001 upwards there are, respectively, ten studies [16], [54], [55], [58], [62], [63], [65], [69]–[71], eleven studies [15], [50]–[53], [59]–[61], [64], [68], [74], and five studies [17], [18], [56], [57], [66]. Three studies [67], [72], [73] did not mention the size of the dataset used.

3.2.3 DL Architectures

All studies proposed at least the implementation of one type of DL architecture. Eleven studies used some form of CNN architecture, such as a typical CNN implementation [16], [18], [53], [55], [58], [67], [70], [71], [73], a Residual Network 50 (a pre-trained CNN architecture) [54], a Capsule NN and a Visual Geometry Group 16 (a pre-trained CNN architecture) [16], and a Deep CNN [60].

Seven studies used variations of NNs, namely a Deep NN [17], [56], [57], a simple NN [63], a Back Propagation NN [69], [74] and a Recurrent NN [72].

Three of the studies used some variation of the LSTM architecture implementation such as a multi-stack deep Bi-directional-LSTM [50], a Bi-directional LSTM [52], a Fast Fisher Vector Bidirectional LSTM [51], a Uni-directional LSTM [52], a Deep LSTM [52], and a Hybrid Bidirectional Unidirectional [52].

One study [59] used 5 variations of the Extreme Learning Machine (ELM) architecture: one simple ELM structure, one Kernel-based ELM, one Multi-Layer ELM, one Multi-Layer Kernel-based ELM, and one Multi-Layer-Perceptron based ELM.

Four studies used another DL architectures, namely a Multi-Layer Perceptron (MLP) architecture [15], [64], an Incremental Learning [62], and a Spatial-Temporal Graph Convolutional Network [68].

Six studies combined more than one architecture into a single pipeline, forming the CNN-LSTM [60], [65], [66], CNN-Support Vector Machine (SVM) [58], LSTM-Recurrent NN [62], and LSTM-Recurrent NN with k-Nearest Neighbours [61] approaches.

3.2.4 Evaluation metrics

The evaluation metrics are an important part of every model pipeline. As the model is developed and trained, several types of metrics are used to measure the quality of the model to understand the performance obtained.

Looking at Table 3.1, it is possible to verify that there are multiple ways to evaluate the performance of the model developed, namely: accuracy (26 studies [15]–[18], [50]–[67], [69]–[71], [74]), f1-score (4 studies [51], [53], [54], [66]), precision (3 studies [51], [53], [66]), recall (3 studies [51], [53], [66]), loss value (3 studies [62], [64], [74]), run time (the training time plus the testing time) (3 studies [53],

[59], [60]), testing time (2 studies [52], [60]), sensitivity (2 studies [50], [61]), specificity (2 studies [50], [61]) and error rate (2 studies [65], [73]).

Single studies also referred to FowIkes-Mallows Index [50], Mathew Correlation Coefficient [50], Bookmaker Informedness [50], Jaccard Similarity Index [50], False Positives [54], Area Under the Curve [66], Receiver Operating Characteristic [66], training time [62], False acceptance rate [70], False rejection rate [70], word error rate [68], and recognition rate [72].

3.2.5 Solution Validation

The solution validation is the stage where the overall performance of a solution is evaluated, through its results with new data, with equal or identical conditions to those where it will be used. At this point, it is intended not only to assess whether the solution meets the needs for which it was created (e.g., operability, effectiveness), but also whether it satisfies the end-users (e.g., acceptability, usability). Of all the twenty-nine studies, only six performed some kind of solution validation [15], [18], [53], [56], [57], [63].

Ameur et al. [15] developed a simple GUI to manipulate Digital Imaging and Communications in Medicine (DICOM) images through the LM camera and the model they developed. The interface comprises four display zones to visualize: the hand skeleton tracked by the LM camera sensor, the original DICOM image, a text displaying the recognized gesture, and the DICOM image after the execution of the command indicated by the gesture. An average of 2 out of 10 repetitions failed.

Brock et al. [53] conducted a preliminary study to assess the user's impression of the proposed framework. The authors asked ten volunteers to freely play Rock-Paper-Scissors (the developed solution) for five minutes each under the two created models (CNN and Random Forest). The order in which they played each of the models was randomized so that there would be no order-based bias. After completing the games for both models, the volunteers answered three different questionnaires. The first form consisted of one to three simple and short assessments (Question 1: Did you feel a difference between the two games?; Question 2: If yes, which of the games did you prefer?; Question 3: If yes, what kind of difference did you experience?), with a corresponding set of possible answer selections (Answer 1: Yes or No; Answer 2: Game 1 or Game 2; Answer 3: Robot response time, Robot response accuracy, Fun of interaction, Other (please specify)), to determine whether users show a preference towards any of the system implementations. The second poll consisted of four quantitative statements, rated on a scale of 1 to 5 (Likert scale), to better understand the user's attitude and feelings toward the proposed game pipeline. The proposed statements were: "I think the overall game is fun", "I think the flow of the

interaction is smooth", "I like the design of the robot reactions", and "If I had a robot, I could imagine playing rock-paper-scissors with it in my free time". The last evaluation queried the participants about their opinion on potential points of improvement, with a short multi-selection field including the following options: robot response time, robot response accuracy, the length of animation (choice 1: make it shorter, choice 2: make it longer), and other issues specified by the participant in written form.

In [57] and [56], the method of evaluation was the same, since they refer to the same study, conducted by the same authors. To evaluate the solution, Hu et al. [56], [57] performed real-time drone control several times, with the developed hand gesture recognition solution, at two locations on the University Campus. All the designed gestures were tested and despite environmental changes such as wind speed, GPS signal, and brightness where the LM device is placed, the authors considered that the system can be used to control real engineering targets.

Lin et al. [63] invited ten volunteers to leave a 2D signature on a piece of paper and ten 3D signatures using the proposed solution. In addition, they were also asked to forge the signatures of other participants so that they could test whether the model could successfully reject intruders by correctly identifying the legitimate user.

Tripathy et al. [18] do not specify the validation protocol used, stating only that the application was tried and evaluated by ten subjects, with no prior experience on how to interact with the LM sensor, and that the system worked.

3.3 Discussion

Of the studies analysed, sign language stands out as the most researched. This may be due to a communication barrier that falls when a good solution is achieved in this field. Given the number of languages (signs or not), and the variation of signs within a sign language, this solution allows deaf people to easily communicate with other people (non-deaf or deaf with different languages), without the need for the latter to learn sign language or to use a translator. The versatility and precision of the LM device can bring more quality to operating room procedures by reducing the risk of contamination during surgical procedures [16] and by reducing the time losses during a medical procedure [15]. Despite these advantages in the medical area, only 4 studies make use of it [15], [16], [66], [74]. This might be because implementing something new in the medical field is hard due to the delicate nature of the area. Innovation is subject to a lot of tests and bureaucracy that can take years. Despite the complications, there is still room for improvement in processes not directly related to the surgery, such as peripheral substitution for application control, and there is a lack of work on this front.

Most of the reviewed studies use a feature-based dataset. This preference can be explained by the use of LM as a criterion for all studies. The amount of information that the device generates is very useful and can probably help achieve more performance than an image [62]. All of these different data points provide researchers with many options to explore, as they need to evaluate and understand which joints give the most value to the problem and which ones create noise, to create an efficient and accurate model. The three image-based approaches explored by [16], [18], [55], despite not making use of the extracted features, still take advantage of LM, since the device uses the infrared range to eliminate lighting and background differences in the final results [55]. In terms of size, the most common is to have less than 10 thousand samples. This is probably due to the difficulty of creating a large and diverse amount of samples in an environment close to the expected conditions of the solution to be deployed.

Of the models used in the studies reviewed, there seems to be a preference for NN architectures, with CNN being the most used type of NN architecture. This may be due to the fact that CNN is suitable for forecasting time-series, as it offers dilated convolutions, which are important for better understanding the relationships between different observations in the time-series [75], and due to the fact that NN with RNNs, in particular, make efficient use of temporal information for both classification and prediction [76], which is good for recognizing dynamic gestures. LSTM architectures are also used for time-series problems but when used in a simple network it has a weak learning ability and fall easily into over-fitting, and when used in a deeper network the recognition rate does not improve significantly with a high sample feature [50], which is probably the reason why many studies do not use this architecture. The remaining studies present a great diversity in the architectures they use, due to the various fields in which hand gesture recognition is being applied, requiring different approaches, but also because researchers try many different things to find the one that can improve performance the most.

Accuracy is the most used metric. Every study intends to develop a model that can recognize hand gestures and the simplest and most intuitive evaluation measure for classifiers is to count the number of mistakes it made out of the total samples to predict. For this metric not to be misleading, it is important to ensure that the dataset used is balanced [77]. The remaining metrics are used to obtain more detailed information about the target and may be required depending on the problem being addressed. Classification times, for example, are necessary if the solution is to be applied in a real-time scenario [78] where fast gesture recognition is required, such as in the field of sign language and the control of non-tactile interfaces.

Most of the studies do not perform any validation of the developed solution. There is no consensus in the literature on what is the best approach to evaluate the system created, which may be a reason for

many studies not using one, and a reason for the diversity of validation solutions found in studies trying an approach. The lack of systems validation may also be due to the extra work required to gather volunteers, develop validation protocols, and select usability questionnaires. One study [53] uses questionnaires but as they were created by the authors it is impossible to compare them with other studies. This gap in the literature presents major limitations, since although the models show good results when tested under conditions outside of laboratory control, they may perform quite differently from where they were trained. In situations where multiple models developed achieved similar accuracies, it is possible to create a real-life test. This test involves using both models to gather feedback from users about how they feel and how responsive their experiences are. Similar to the approach taken by Brock et al. [53], this can help determine which model performs better in practical, real-world situations.

3.4 Conclusions

This review focused on DL-based hand gesture recognition with the LM device and aimed to analyse the current literature by extracting the relevant specifications for algorithm development (such as the study focus, dataset type and size, DL architectures, evaluation metrics, and solution validation) to answer the following RQ: What specifications should be considered for the development of a LM hand gesture recognition application fitted for surgical navigation? The optimal solution should consider technological and clinical requirements. Thus, for the final solution to be suitable for a real-world application, it should consider the following technological requirements: 1) ensure a balanced dataset; 2) the model architecture should be defined based on the required gestures (static or dynamic); 3) define the number and which gestures are required; and 4) define a validation protocol that includes real-life situations with the end-users. This set of requirements should help guarantee that the overall solution created can support the needs of an operating room, with little room for error. Considering the clinical requirements, the final solution should: 1) include end-user-centred gestures, so that the end-user can manipulate digital medical images and 3D anatomical models as easily and intuitively as possible; 2) define a validation protocol to study the usability of the application. These specifications should make it easier for surgeons to adopt these tools, as they feel they can do everything they were doing before, successfully.

4. SOLUTION DESCRIPTION

The main outcome of this dissertation was a hand gesture recognition tool designed to control a GUI without the need for physical contact via a keyboard and/or mouse, which would reduce the risk of contamination during surgery and the time needed to sterilize peripherals. The development of this work comprised the construction of a large dataset that was later fed into AI models for training purposes. It also involved the creation of an application that used the trained model to infer a hand gesture in real time and perform the corresponding mouse action accordingly. This chapter describes the proposed methodology, the hardware and the software required throughout the development of the tools needed to achieve all the goals, and defines the actions required to replace the computer mouse.

4.1 Methodology

The work carried out was divided into a series of smaller steps, ensuring that the technical and clinical requirements defined at the end of the previous chapter were met. The first step was to understand which actions are normally performed by a conventional mouse to control a surgical navigation software (e.g., click, scroll). Since each action is associated with a different hand gesture to avoid conflicts, the number of actions/gestures required was quickly defined, fulfilling the third technological requirement. To design an end-user-centred tool, the hand gestures themselves were defined by the end-users achieving the first clinical requirement, through an experimental protocol carried out at the *Trofa Saúde Braga Centro* Hospital, to determine which hand gestures were the most intuitive, the easiest and the most suitable in an operating room environment for controlling a surgical navigation software. An application was created to help with the protocol intervention process.

The following step was to create a dataset with the defined set of hand gestures in order to have a large collection of information to feed the AI models. This step involved the creation of another experimental protocol. Once the dataset had been created and analysed to ensure that it was balanced, fulfilling the first technological requirement, several AI models were trained to find the best and most appropriate architecture for the defined gestures, meeting the second technological requirement. Then, using the best model selected for real-time inference, an application was developed to control the surgical navigation software using only hand gestures. As this is a time-sensitive task, performance is an important factor to ensure.

Finally, the last step consisted of a laboratory validation and a final validation with end users of the final solution, to understand the overall quality of the programme, its usability and efficiency, and how it

compares to a conventional mouse interaction with the surgical navigation software. To do this, an experimental protocol was created with a set of actions to be performed. Users were then asked to respond to a usability test so that the difference between a typical interaction and the one could be measured by a standardized test and used as a reference point for future work. This fulfils the fourth technological requirement, the definition of a validation protocol including real-life situations with the end-users, and the second clinical requirement, the definition of a validation protocol to study the usability of the application.

4.2 Hardware

This subsection presents the hardware used during the development of this work. It provides a detailed description of the LM device used to collect the hand gesture information and its capabilities, as well as the specifications of the computer used to collect the information, develop the programs and train the AI models.

4.2.1 Leap Motion

The LM controller is an optical hand-tracking module that tracks hand movements. It is composed of two 640x240-pixel near-infrared cameras spaced 40 millimetres apart and three LEDs (one between the two cameras and the other two at the tips of the device) that track the infrared light at a wavelength of 850 nanometres. The device has a 140x120° field of view and the interaction zone has a depth of up to 80cm, although 60cm is preferred [79]. A physical and schematic view of the device is illustrated in Figure 4.1.

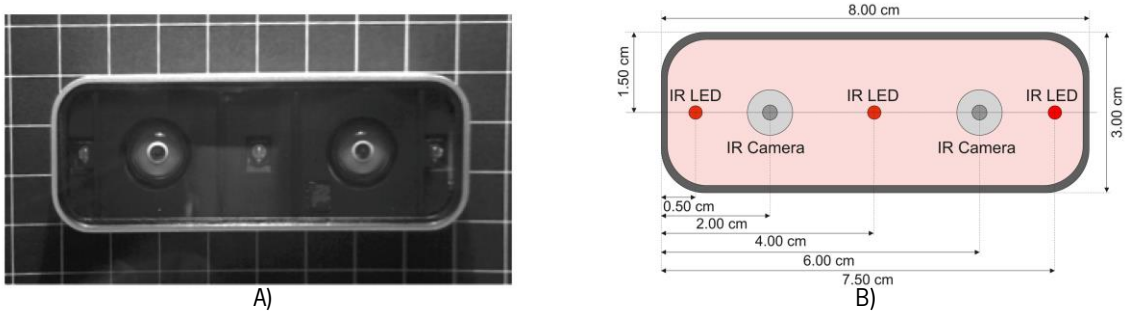


Figure 4.1 – LM Controller: A) Physical view, B) Schematic view. [80].

To get the most performance out of this hardware, the device comes with the LM API and the LM service. The first provides the developer with the resources to access the device, change settings and retrieve the information. The latter sits between the hardware and any application developed and is responsible for processing the information generated before sending it [81].

The LM employs a cartesian coordinate system with the origin centred at the top of the device. As shown in Figure 4.2, the x-axis is parallel to the device and the z-axis is perpendicular to the x-axis, pointing to the right of the user and towards the user, respectively. In the vertical plane, the device has the y-axis pointing upwards [82].

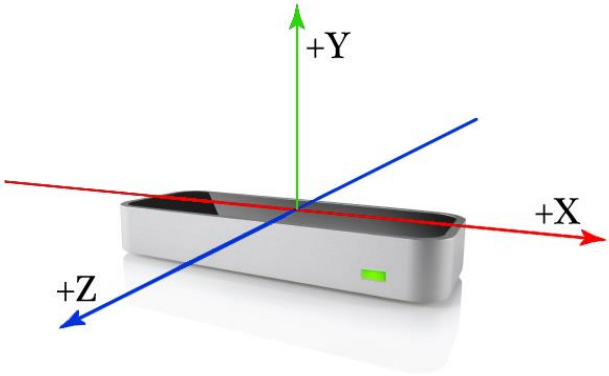


Figure 4.2 – LM cartesian coordinate system [82].

The information collected by LM arrives at the application in the form of a Frame object, which contains either one or two instances of a Hand object. For each hand, it is possible to extract information about position, direction, velocity, and rotation. It is also possible to know whether each finger is extended and the position, direction, and width of each one of the four bones in the finger. As depicted in Figure 4.3, all fingers except the thumb contain four bones, but for simplicity the API maintains the four bones for the thumb but gives the metacarpal a default value of 0 [82].

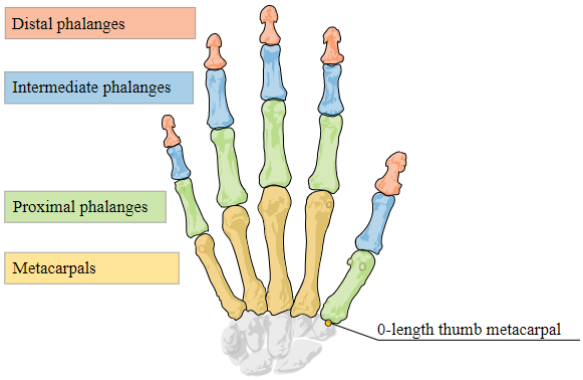


Figure 4.3 – Bones available in the finger object [82].

Table 4.1 shows the features provided by LM in more detail and the corresponding values. The device can also take grayscale pictures at a variable speed, and the LM service also removes some of the noise from the picture by removing background images and ambient lighting, so that the visible hands are in focus. Despite all the qualities of the LM device, there is one limitation that should be kept in mind. As with some vision-based tracking systems, the tracking system can fail if part of the hand is occluded.

Its service has an algorithm that tries to infer the occluded positions, but there is no guarantee that it will do this correctly. The same issue occurs when one hand is on top of the other [83].

Table 4.1 – Features provided by the LM service

Body part	Feature	Values
Hand	Type	Left/Right
	Pinch strength	[0,1]
	Pinch distance	value ≥ 0
	Grab strength	[0,1]
	Grab angle	Rad
	Visible time	Microseconds
Palm	Direction	Vector(x,y,z)
	Normal	Vector(x,y,z)
	Orientation	Quaternion(x,y,z,w)
	Position	Vector(x,y,z)
	Velocity	Vector(x,y,z)
	Width	Millimetres
Finger	Is extended	1 or 0
Bone	Rotation	Quaternion(x,y,z,w)
	Base	Vector(x,y,z)
	Tip	Vector(x,y,z)
	Width	Millimetres
Arm	Rotation	Quaternion(x,y,z,w)
	Prev_joint	Vector(x,y,z)
	Next_joint	Vector(x,y,z)
	Width	Millimetres

4.2.2 Computer Specifications

The LM has some recommended system requirements (Table 4.2) that the host computer must meet to take full advantage of the specifications. An ASUS computer with an Intel Core i7-8750H and an NVIDIA GeForce GTX 1050 Ti GPU was used to run and connect the software required to build the tool. According to Table 4.2, the computer comprises all the recommended requirements for using the LM device.

Table 4.2 – Recommended system requirements for using LM and specifications of the computer used

Component	Recommended system requirements	Computer used
Processor	AMD Phenom II/Intel Core i3/i5/i7 equivalent or better	Intel Core i7-8750H
GPU	Not applicable	NVIDIA GeForce GTX 1050 Ti
Memory	2 GB RAM or more	16 GB
USB port	1x USB 2.0 or newer	3x USB 3.0
Operating system	Windows 7+ / Mac OS X 10.7	Windows 11

Although there are no minimum specifications for using TensorFlow, or for training AI models in general, the more powerful the computer, the less time is needed for the training process. The specifications of the computer used can handle the amount of work involved in this project, but it does take some time and does not allow for a very complex architecture.

4.3 Software

Throughout this dissertation, several pieces of software were developed with different objectives to carry out the steps required to complete the agreed-upon work. The following subsection gives a brief description of each of the main software programs used, including packages from programming languages, their purpose in this project, and the motive behind their choice.

4.3.1 Visual Studio Community 2022

Visual Studio Community is an Integrated Development Environment that allows the user to edit, debug, build and publish code. It gives the developer a plethora of functionalities, namely, graphical designers, support for C++, C#, .NET and several other programming languages, git source control and a built-in packet manager to facilitate the import of external libraries. This tool was used to develop an application that could connect to the LM Service and retrieve the feature and image data necessary to build the dataset. This Integrated Development Environment was chosen due to a combination of factors: the need to develop a GUI, the fact that the LM Service only supports C++, and the ease of implementation of C++ GUI applications.

4.3.2 Visual Studio Code

Visual Studio Code is a lightweight but powerful source code editor that allows users to edit, debug, build, and publish code. It has native support for some programming languages, namely, JavaScript, TypeScript and Node.js, and thanks to a rich ecosystem of extensions, it supports other languages and runtimes, such as C++, C#, Java, Python, PHP, Go and .NET. This editor was used with two different applications in mind: the development of a Python script that can be used in conjunction with the TensorFlow library to train AI models, and the development of a Python application that uses the Open NN Exchange runtime to perform real-time inference and uses PyAutoGUI to control the computer. This editor was used because it is lightweight and has an easy integration with the Python language.

4.3.3 TensorFlow, CUDA/cuDNN

TensorFlow is an end-to-end open-source platform developed by Google for ML, that makes it easy for beginners to venture into the world of AI and for experts to create complex models to solve hard challenges. TensorFlow's capabilities can be enhanced with CUDA, an API developed by NVIDIA that provides a Toolkit for creating high-performance GPU-accelerated applications, and the cuDNN library, which complements the CUDA API with highly tuned implementations commonly used in deep NNs, such as convolution, pooling, normalization, and activation layers. These are used together to significantly reduce the training times of the models and increase the developer's ability to test different and more complex architectures. This combination of software was chosen due to the availability of an NVIDIA graphics card and the author's familiarity with TensorFlow.

4.3.4 PyAutoGUI

PyAutoGUI is a package present in the Python Package Index that gives the user the ability to control the mouse and keyboard from a script and automate interactions with other applications. It has support for Windows, MacOS and Linux, making it possible to move the script between different platforms without any changes in the code. This package is used in the final application developed to match the predicted gesture with an action on the screen. This tool was chosen for its portability, which makes the program easier to maintain, its ease of use and its performance.

4.3.5 Open Neural Network Exchange

ONNX is an open standard for ML interoperability. It allows the developers to use models developed and trained in other ML tools such as TensorFlow and PyTorch. The ONNX runtime simplifies hardware access and optimizes models for inference time. As the final application is based on real-time gesture recognition, inference time is critical, and the use of this tool is vital to the performance of the solution.

4.3.6 Pandas

Pandas is a fast, powerful, and easy-to-use open-source data analysis and manipulation tool built for Python. This package allows the user to read data structures stored in file format and convert them into an Excel-like table called a DataFrame. This DataFrame object has a set of tools that allow easy manipulation of the data, either to change rows, columns, and the shape of the data, or simply to get some statistics from the data, such as the normal distribution, averages, and others. This tool was mainly used to clean and prepare the collected data to be used in the training process of the developed AI models. It was chosen because of its ease of implementation, performance, and integration with Python.

4.3.7 Scikit-learn

Scikit-learn is an open-source ML library that provides several model fitting algorithms like SVM, Binary Tree, and Random Forest. It also provides a wide set of tools for data processing, model selection, and model evaluation, such as confusion matrix, f1-score, and accuracy score. This tool was used to develop ML models with the generated dataset created because of its integration with the Python environment, its ease of implementation and experience with the tool.

4.4 Definition of the actions required to replace the computer mouse

The main purpose of this dissertation is to replace the use of the mouse to control the computer with a hand gesture-based approach using LM. To achieve this, it is important to first consider what actions are currently performed with the mouse, to then translate them into the gestures necessary to keep the functionality of the applications.

As this dissertation focuses on the control of surgical navigation software to reduce the need for sterilization of equipment between surgeries, the NavPi surgical navigation application, developed by BiRDLab team, will be the subject of evaluation for the necessary actions on the program. NavPi is intended to be used during surgical procedures, allowing visualization of 2D images and 3D models, and

analysis of medical information as patient's clinical data. After a thorough analysis of the software, the actions needed to explore the full functionality of the application were clear. These are:

- Click action
- Zoom in action
- Zoom out action
- Move action
- Rotate action
- Cursor action

With the number of actions established, it is possible to determine the number of gestures required to control the LM application in a surgical environment, which is 6. It is then important to define which gestures should be used. The RQ 1, answered in Chapter 3, stated that the defined hand gestures should be user-centred, so that the end-users can control surgical navigation software, enabling them to manipulate digital medical images and 3D anatomical models as easily and intuitively as possible. To achieve this, a protocol was created to define what these gestures would look like. A crucial step was completed by identifying the number of necessary gestures to serve the identified actions.

5. SELECTION OF THE MOST APPROPRIATE HAND GESTURES FOR AN END-USER-CENTRED APPROACH

This chapter describes the experimental protocol used in an initial stage of the work to define the hand gesture to be used. This protocol followed an experimental study design aiming to understand, with several surgeons, the most intuitive and suitable hand gestures, for a given set of actions, to manipulate a surgical navigation software application during a surgical procedure using LM. First, the participants and their characteristics are specified. Then, the intervention is explained, followed by the data collection, carried out to achieve the specified purpose. Finally, the results are presented and discussed, and some conclusions are made.

5.1 Participants

The study was approved by the institutional review board of the *Trofa Saúde*. All participants filled out an informed consent, based on the Helsinki declaration and the Oviedo convention, to participate in this study. Four participants, all surgeons, fulfilling the KPI of a minimum of 3 surgeons defined on objective 2 of this dissertation (gender: 4 males and 0 females, age: 40.25 +- 6.38 years, dominant hand: 0 left and 4 right, previous experience with touchless controls: 4 with no experience) (Table 5.1), recruited and admitted in the *Trofa Saúde Braga Centro* hospital, were enrolled in the study according to the inclusion criteria of being an orthopaedic surgeon. There were no exclusion criteria. The previous experience with touchless controls was asked to see if there was any correlation between the gestures proposed by the participants and the gestures used to control the touchless tool. As no one had previous experience, no conclusions could be drawn.

Table 5.1 – Age (years), gender (male/female), dominant hand (left/right) and previous experience with touchless controls of each participant for the selection of the most appropriate hand gestures

Participant	**Age (years)**	**Gender**	**Dominant Hand**	**Previous experience with touchless controls**	
				Yes/No	**Touchless Tools**
Participant 1	39	Male	Right	No	-
Participant 2	36	Male	Right	No	-
Participant 3	35	Male	Right	No	-
Participant 4	51	Male	Right	No	-

5.2 Intervention protocol

The experimental study consisted in the following two phases: familiarization phase and data acquisition phase. The familiarization phase consisted of showing the participants the interface and letting them play with the tool, so that they become familiar with the process. Data acquisition phase included two parts: first, with hand gestures proposed by the end-user and second, suggested by the author of this dissertation. In each part of the data acquisition phase, a set of 5 questions were presented to the participants, each corresponding to one of the identified required actions: 1) What hand gesture would you make to click on a button on the screen?; 2) What hand gesture would you make to move a 3D model of a bone from the left side of the screen to the right side of the screen?; 3) What hand gesture would you make to rotate a 3D model of a bone to the left?; 4) What hand gesture would you make to increase the zoom in a 3D model?; and 5) What hand gesture would you make to decrease the zoom in a 3D model? In the first part, in which the end-users were asked to propose the hand gesture that they felt more intuitive for each question, the order of the questions was randomized to avoid carry-out time effect. In the second part, participants were asked to perform a pre-defined suggested gesture for each question. These pre-defined suggestions have been created to provide end users with a different alternative that may be less technically complex than the one they proposed. For each part, one trial is performed per question. The pre-defined gestures are shown in Figure 5.1. At the end of each suggested gesture, the participants were asked if they preferred their own proposed hand gesture or the one suggested by the author of this dissertation. This intervention takes 12 minutes to complete.

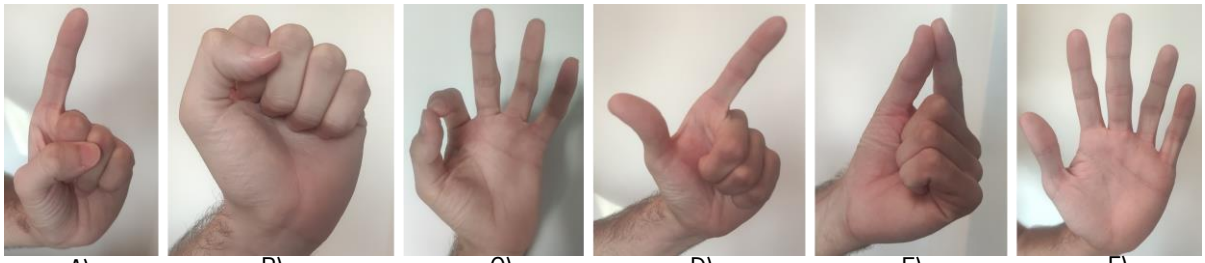


Figure 5.1 – Hand gesture suggested by the author for: (A) click, (B) rotate, (C) move, (D) zoom in, (E) zoom out, (F) cursor.

There were several reasons behind the decision of these hand gestures. The first one was based on the use case of the final application. If the goal is for the surgeon to use this tool during a surgical procedure, it is important that the functionality of the program is not reduced when the surgeon is holding a tool. For this reason, all hand gestures are one-handed. With this in mind, the gestures were then based on literature, software implementations, and intuitiveness between the gesture and the desired action.

Finally, all gestures should be distinctive enough to be better recognized by the AI models. At last, the gesture to control the cursor was chosen so that it wouldn't be the same as any of the other five gestures. Furthermore, because it should be the most used gesture, it was important that it was not only simple and intuitive, but also basic and comfortable.

An application was developed to support the intervention process and to guide the participants through the protocol. The application has four basic functionalities, as shown in Figure 5.2. The first one starts the collection of the information (features and images) in a fixed window of 3 seconds in which the participant is supposed to perform the gesture (A). The second button proceeds to the next question (B). The third gives a visual demonstration of the action being performed on the surgical navigation software with the mouse (C). The fourth is simply a button to reset the whole process to be ready for a new participant (D).

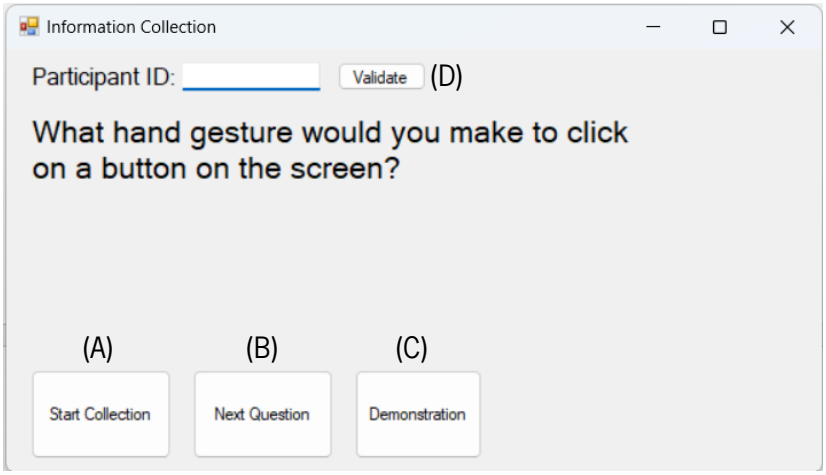


Figure 5.2 - GUI used in the collection of the most intuitive hand gestures using LM.

All four participants completed the entire protocol. During the intervention, the main outcome and information collected is the participant's gesture preference. The LM camera also captured images to document the participant's thought gestures. To protect the privacy of the participants, the anonymity of their opinions and answers was guaranteed.

5.3 Results and discussion

Table 5.2 presents the participants' gesture preference for each question: original, if the participant prefers the gesture proposed by himself; suggested, if the participant prefers the gesture proposed by the author of this dissertation. For cases where the volunteer performed the same or very similar hand gesture to the one suggested by the author, the table refers to it as similar. Since the gesture was the same, it will be counted as the suggested gesture for counting purposes.

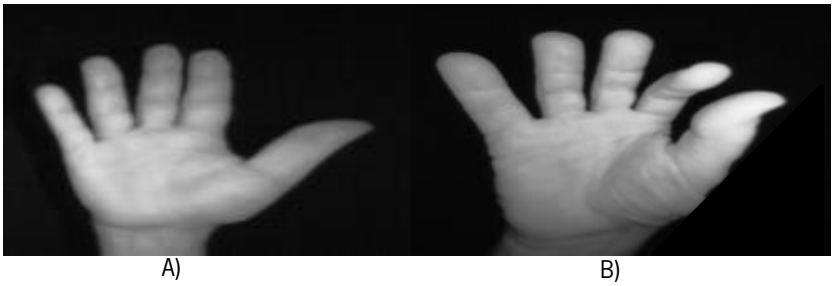
Table 5.2 – Participant's hand gesture preference

Participant	Click	Move	Rotate	Zoom in	Zoom out
Participant 1	Original	Suggested	Suggested	Similar	Similar
Participant 2	Suggested	Original	Original	Similar	Similar
Participant 3	Suggested	Original	Suggested	Similar	Similar
Participant 4	Similar	Suggested	Original	Similar	Similar

For the click action, two participants preferred the suggested gesture to their own, one performed a similar gesture to the one suggested by the author, and one participant felt his proposed hand gesture was more intuitive. As it can be observed on Figure 5.3, this participant proposed pointing with the index and middle fingers, whereas the suggested gesture by the author was only to point with the index finger. This is quite similar and as the majority preferred the one suggested by the author, the latter will be the one used to train the models.

Figure 5.3 – Gesture proposed by participant 1 for the click action.

For the move action, two participants preferred their proposed hand gesture and the other two preferred the suggested gesture by the author. The proposed hand gestures can be seen on Figure 5.4. Since the two subjects who preferred their own gesture performed different gestures (one had the hand fully open while moving left/right and the other pinched the thumb and index finger while moving left/right), the majority will prevail, meaning that the gesture suggested by the author will be the one used to train the models.



A) B)

Figure 5.4 – Gesture proposed for the move action (A) by participant 2. (B) by participant 3.

For the rotate action, two participants preferred the gesture suggested by the author and the other two preferred their own (one had the hand completely closed while rotating the wrist and the other had the hand open as if holding a can while rotating the wrist) The gestures proposed by the participants can be seen on Figure 5.5. This action exhibited the same behaviour as the move action and for the same reason, the suggested gesture will be the one used to train the models.

A) B)

Figure 5.5 – Gestures proposed for the rotate action (A) by participant 2. (B) by participant 4.

For the zoom in and zoom out actions, all the participants performed an equal or very similar gesture to the one suggested by the author. This gives confidence that the gestures decided to perform the zoom in and zoom out actions are the most intuitive.

5.4 Conclusions

This section aimed to understand, together with the surgeons (end-users), which gestures would be the most intuitive to perform the set of actions defined in the previous chapter to control the surgical navigation software. At the end of the intervention, some variability in gesture preference was observed. For the click gesture, two participants preferred the gesture suggested by the author, one performed the same gesture, and one preferred the gesture idealized by him. For the move gesture and rotate gesture, two participants preferred the gestures suggested by the author, and two participants preferred the gesture idealized by them. For the zoom in and zoom out gestures, all participants performed the same gestures as the ones suggested by the author. It was also noticeable that without any instruction to do so, all participants only used one hand to suggest their own gestures, indicating that one hand gestures are likely the correct direction. Ultimately, the gestures suggested by the author were the ones decided to be used to control the surgical navigation software. Nevertheless, the gestures were tested and are in line with the end user, which increases the viability of the tool and achieves a design centred on the end user. These gestures will be used in a later stage of this dissertation to develop a large dataset to feed AI models for automatic hand gesture recognition.

6. SYSTEM DESCRIPTION: AUTOMATIC HAND GESTURE RECOGNITION

This chapter describes the development of the final hand gesture recognition tool using LM. It begins by explaining the protocol used to create two datasets of the defined hand gestures, one with the features of the hands (feature-based) and one with the images from the hands (image-based). The data analysis and processing followed to clean the data for using in the model training process. The architectures explored on the two types of data are then described and explained in detail. Next, the results of the models are presented, analysed, and discussed, benchmarking them against some of the studies analysed in the review in Chapter 3. After selecting the best model, the creation process of the hand gesture recognition tool is explained. At last, some conclusions are made about each section explored in this chapter.

6.1 Dataset creation: feature-based and image-based hand gestures

This section describes the protocol used to create a dataset of hand gestures. This dataset will comprise the gestures defined in Chapter 5. The protocol followed an experimental study design with the aim of creating a large dataset of feature and image-based data using LM. This subsection starts by specifying the participants and their characteristics. Then, the intervention is explained, followed by the data collection that was carried out to achieve the desired result. At last, the data processing and analysis is explored.

6.1.1 Participants

All participants filled out an informed consent, based on the Helsinki declaration and the Oviedo convention, to participate in this data collection. Twenty-one participants (gender: 14 males and 7 females, age: 26.24 ± 3.39 years, dominant hand: 1 left and 20 right) (Table 6.1), recruited and admitted at the University of Minho, were enrolled in the study by voluntary participation, fulfilling the KPI of a minimum of 15 participants defined in the objective 3 for this dissertation. There were no exclusion criteria.

Table 6.1 – Age (years), gender (male/female), dominant hand (left/right) of each participant for the creation of the hand gesture dataset

Participant	Age	Gender	Dominant Hand
Participant 1	25	Female	Right
Participant 2	22	Male	Right
Participant 3	23	Female	Right
Participant 4	22	Male	Right
Participant 5	24	Male	Right
Participant 6	29	Male	Right
Participant 7	22	Male	Right
Participant 8	26	Male	Right
Participant 9	27	Female	Right
Participant 10	30	Female	Right
Participant 11	27	Male	Right
Participant 12	29	Male	Right
Participant 13	30	Female	Right
Participant 14	27	Female	Right
Participant 15	23	Female	Right
Participant 16	23	Male	Left
Participant 17	22	Male	Right
Participant 18	32	Male	Right
Participant 19	26	Male	Right
Participant 20	29	Male	Right
Participant 21	33	Male	Right

6.1.2 Intervention protocol

The intervention begins by explaining to the participants the GUI for the data collection. For example, when a hand gesture is considered valid and the time window (3 seconds) in which the hand gesture is collected. The experimental study consisted of 3 familiarization trials with the LM device, followed by two phases of data collection: first, the participant performs the gestures without surgical gloves, and second, the participant performs the gestures with surgical gloves. The use of surgical gloves is important to ensure that the final application can be used in the context of an operating room but is also ready for use in other situations where gloves are not required. For each phase, 6 valid trials (3 for

each hand) are performed in a row per manipulation action, for the following six actions: a) Zoom in, b) Zoom out, c) Move, d) Rotate, e) Click, and f) Cursor action. Each action was performed within 1 minute and 30 seconds (6 trials) and there was a 1-minute rest period between the two phases.

6.1.3 Data collection

All 21 participants completed the entire protocol. Two sources of information were collected during the intervention: hand gesture features, provided by the calculations of the LM device, and the hand gesture images, captured by the LM camera.

For the hand gesture features, there was a fixed window of 200 frames to collect. In these frames, all features made available by the LM service were collected. For the hand gesture images, since the frame rate of collection cannot be fixed, the number of images collected is the number of images the service is capable of recording within the collection of the 200 feature frames.

6.1.4 Data processing and analysis

The data processing and analysis were carried out using the Python programming language in the Visual Studio Code editor. This step is essential before training the models, as the models are highly dependent on clean data.

The first step was to remove invalid trials of some participants from the dataset. These were caused by several reasons: 1) the LM service did not recognize a hand; 2) the subject switch hands in the middle of a trial; 3) the number of hands being captured by the LM changes in the middle of a trial; 4) the volunteer did not perform the gesture correctly. This step was important to avoid unbalancing the dataset and feeding the models with incorrect or incomplete data.

The second step consisted of checking the data for missing or unexpected values. The data is sent by the LM service, so such occurrences are unlikely, however it was important to guarantee that no error had occurred.

Figure 6.1 presented a diagram of the division of the feature-based dataset for better readability, showing the number of frames per volunteer and how it translates into the total 21 volunteers. Since the number of feature frames collected per trial is fixed by the data collection tool developed (approximately 67 frames per second), each trial collected 200 frames. Thus, 600 frames are collected for each hand in each gesture (3 trials), totalling 1200 frames per gesture. The six defined hand gestures were performed twice (one phase without surgical gloves and another phase with surgical gloves), so that 7200 frames were collected for each phase. The two phases together resulted in 14 400 frames of information per volunteer. Considering all the 21 volunteers, there were 151 200 frames collected for each hand. For

each gesture in each phase, 25 200 frames were collected. In each phase 151 200 frames were collected, totalling 302 400 frames collected.

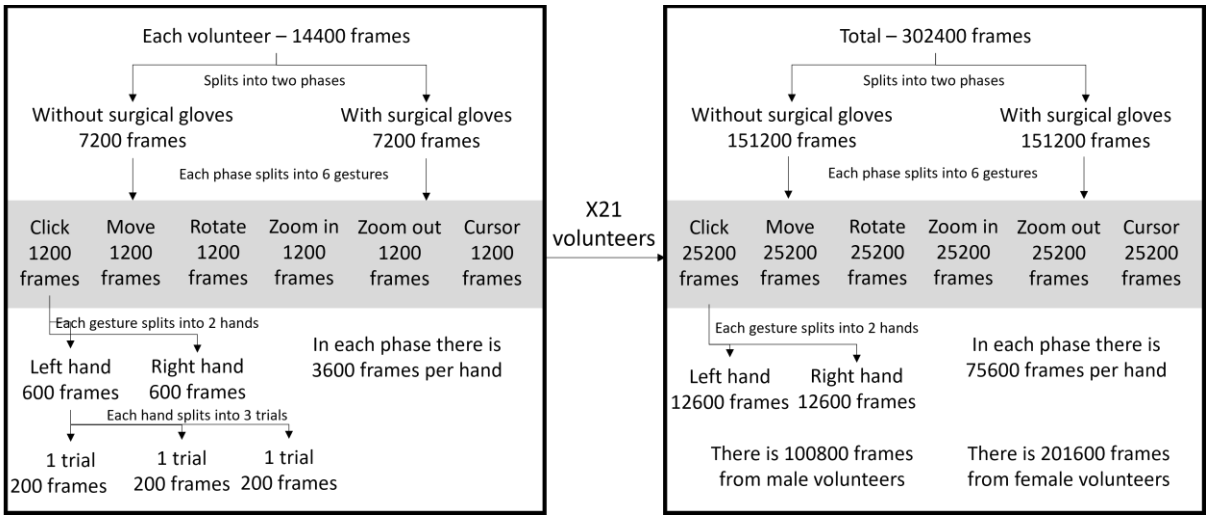


Figure 6.1 – Feature-based hand gesture dataset. Number of frames per participant (left side) and total number of frames among all the participants (right side).

Figure 6.2 shows a diagram of the division of the image-based dataset, showing the number of frames per volunteer and the number of frames across all the volunteers. The frame rate of the images collected by the LM camera cannot be fixed and as such there is some variability from trial to trial. Each trial has an average of 225 ± 7 frames. For each phase (with and without surgical gloves), and in each gesture (3 trials) there is an average of 675 ± 16 frames for the left hand, and an average of 673 ± 18 frames for the right hand. The sum of the two hands results in an average of 1348 ± 31 frames for each gesture performed in each phase. For each phase (with and without surgical gloves), each gesture was performed 6 times (3 times with left hand and 3 times with right hand) resulting in an average of 8078 ± 154 frames collected in the phase without surgical gloves, and an average of 8099 ± 179 frames collected in the phase with surgical gloves. The two phases together gave a total of 16177 ± 330 frames per volunteer. Taking all 21 volunteers into account, there was 170 070 frames with the left hand and 169 644 frames with the right hand. Across both phases, the click gesture had 56 482 frames, the move gesture had 56 736 frames, the rotate gesture had 56447 frames, the zoom in gesture had 56782 frames, the zoom out gesture had 56817 frames and the cursor gesture had 56450 frames. From the phase without surgical gloves, 169 663 frames were collected and, from the phase with surgical gloves, 170 081 frames were collected, totalling 339 714 frames. In any of the two datasets created, the KPI of having more than 200 000 samples established in the objective 3 of this dissertation is fulfilled. This number was based on the biggest dataset found on the literature review.

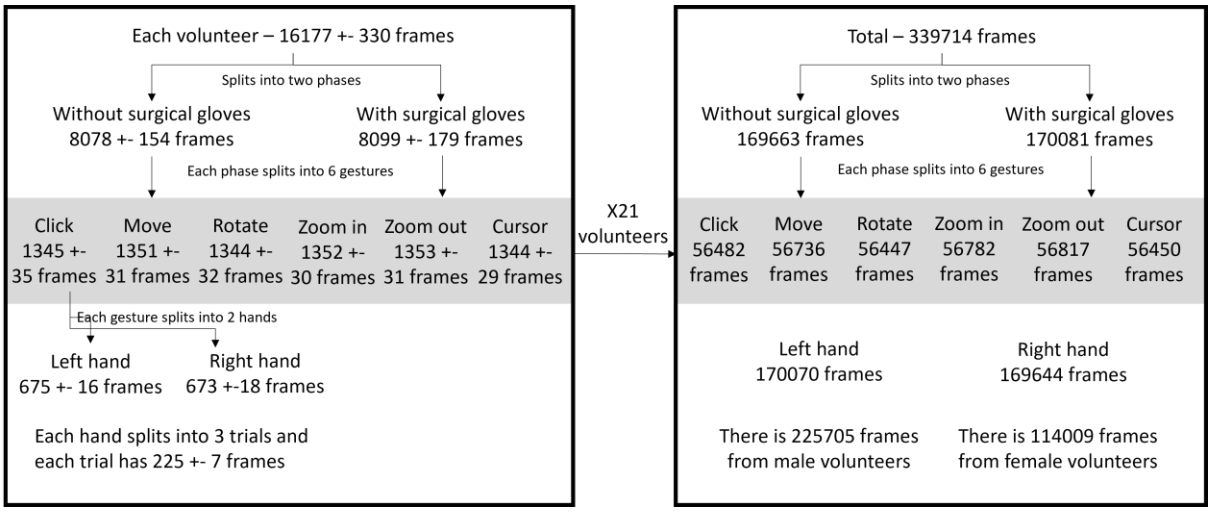


Figure 6.2 – Image-based hand gesture dataset. Number of frames per participant (left side) and total number of frames among all the participants (right side).

After the processing of the dataset, the balance of the dataset was examined to ensure that all classes in the problem were equally represented, so that the model would not have a bias towards one or a set of classes and having a better chance for generalizing the data. Figure 6.3 shows that the developed dataset is balanced in all classes, for both the feature-based hand gesture dataset and the image-based hand gesture dataset.

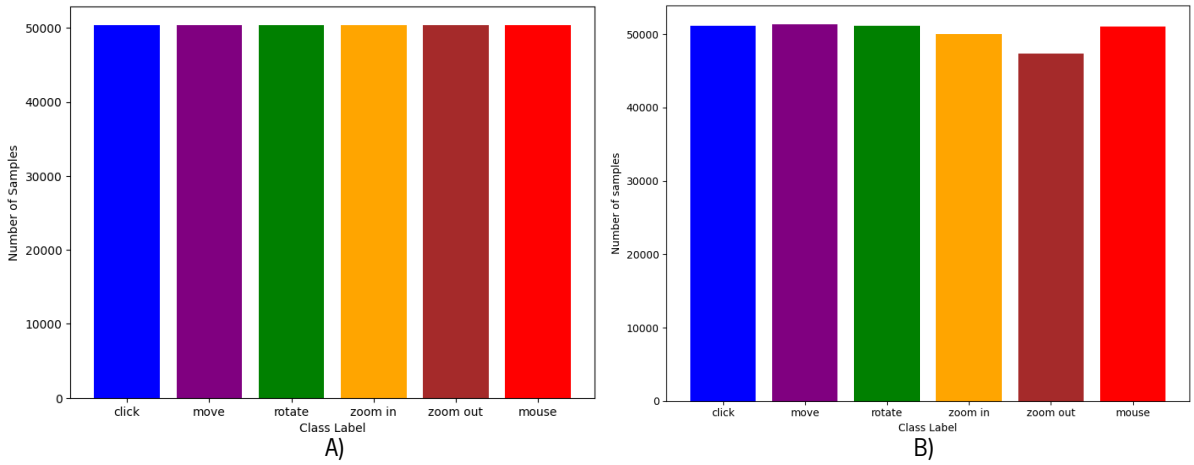


Figure 6.3 – Class distribution of (A) feature-based and (B) image-based hand gesture datasets.

Finally, the class label of the dataset distinguishing gestures with and without surgical gloves was reduced to only represent the six actions to be classified. For the feature-based dataset, the final step was to normalize each feature to give the AI models a better chance of generalizing the information and to minimize the chance of exploding/vanishing gradient problems. This step was also performed on the images but integrated into the architectures.

6.2 AI models architecture

This subsection describes the different AI architectures that were explored throughout the process and the reasoning behind them. Despite the initial focus of this dissertation on the development of DL models, as evidenced by the review done in Chapter 3, to have a more broad and complete approach some ML algorithms were explored to understand if DL models bring advantages and if they are necessary to develop a good hand gesture recognition tool.

Since two datasets were created (feature-based and image-based), different types of architectures were explored for each. The architectures explored for the feature-based hand gesture dataset are presented first, followed by the architectures explored for the image-based hand gesture dataset. Some architectures were chosen based on the literature, the most used for the feature-based datasets were the NNs and the most used for the image-based datasets were the CNNs. Figure 6.4 gives an overview of the pipeline involved in the development of the models.

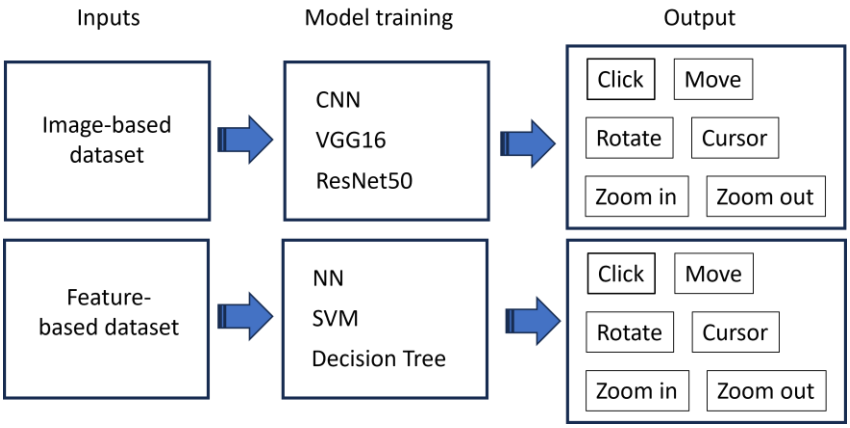


Figure 6.4 – AI model development pipeline.

6.2.1 Feature-based dataset

NN with linear activation

The first model developed for the hand features follows a simple NN architecture and it was based on the article written by Hu Bin [57]. This architecture consists of four dense layers with linear activation function and 200, 100, 60 and 30 nodes, respectively, followed by a final dense layer with 6 nodes (one for each class) and a softmax activation function to attribute a probability to each of the classes available. Figure 6.5 presents a diagram of the architecture.

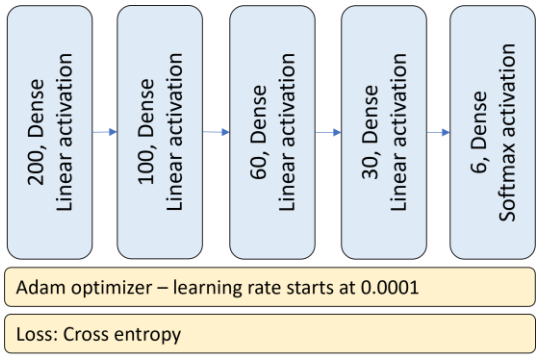


Figure 6.5 – NN architecture with linear activation.

NN with Rectified Linear Unit activation

The second model developed for the hand features follows a similar NN architecture to the first model and it was based on the article written by Li Xichao et al. [17]. The difference is the change from linear activation on the four dense layers to a Rectified Linear Unit (ReLU) activation. Otherwise, it also has four dense layers with 200, 100, 60 and 30 nodes, respectively, and a final dense layer with 6 nodes (one for each class) and a softmax activation function. Figure 6.6 shows a diagram of the architecture.

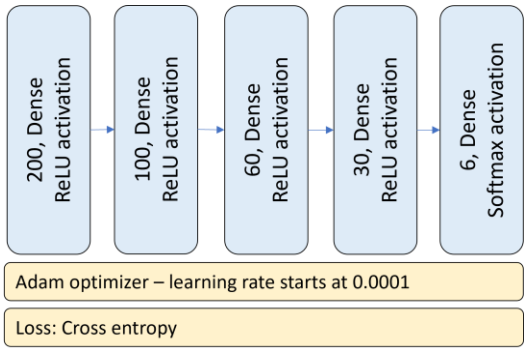


Figure 6.6 – NN architecture with ReLU activation.

SVM

The third model developed follows a SVM algorithm. SVM is an interesting approach to accurately separate the data points from the 6 gesture classes using a hyperplane. As discussed in Chapter 2, one of the disadvantages of ML over DL is the inability to extract the relevant features from the dataset. As this is a ML model, it is important to reduce the features of the dataset. The main differences between the click, rotate, zoom in and zoom out gestures can be described by the palm, thumb, index, and middle fingers positions and by the distance between the tip of the thumb and the tip of the index fingers, so the spatial coordinates (x,y,z) of the palm, thumb, index and middle fingers and the pinch distance were

extracted. To distinguish between the move and mouse control gestures, the grab strength was extracted. This combination of features was based on the features decided in some studies reviewed in Chapter 3 [56], [61], with some adjustments due to the difference in gestures.

Decision Tree

The fourth model developed also comes from a ML algorithm. The Decision Tree was also considered, as it was deemed interesting that the model could successfully classify the 6 gestures by following a simple set of decisions. The same reasoning of feature selection was followed for this algorithm.

6.2.2 Image-based dataset

CNN with average pooling

The first model developed had a simple CNN architecture and it was based on Enikeev et al. [55]. This choice was made due to the proven capabilities of CNN in predicting images, and the good results achieved with this architecture.

This architecture starts with a Rescaling layer that normalizes every pixel in the image, converting the [0,255] range into a [0,1] range. This is then passed to a convolutional layer with 14 filters and a 5x5 kernel that slides through the image in a 1x1 stride. The output of this layer is then passed to the ReLU activation function, which passes the values to the pooling layer with a size of 2x2 and a stride of 2. This down-samples the input by taking the average of each input window. A dropout layer with a 0.25 rate is then added, dropping 25% of the connections. This convolution-pooling-dropout combination is repeated four times. The data then passes through a layer that flattens the multi-dimensional array into a single-dimensional array to then feed to the final classification layer with 6 nodes (one for each class). The diagram of the architecture is shown in Figure 6.7.

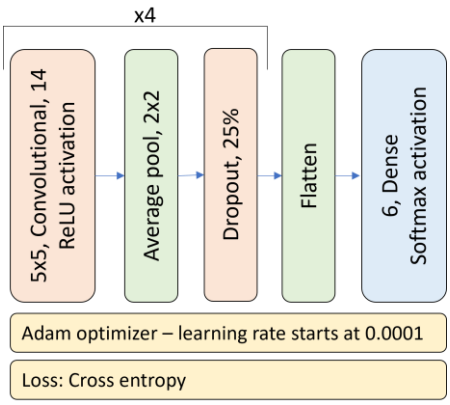


Figure 6.7 – CNN with average pooling architecture.

CNN with max pooling without dropout layers

The second image-based model developed also had a simple CNN architecture and was based on Tripathy et al. [18]. In his study, the model achieved good results using max pooling layer, so it should be interesting to see what effect this might have on the performance of the dataset created in this dissertation.

This architecture starts with a rescaling layer that normalizes every pixel in the image, converting the [0,255] range to a [0,1] range. This is then passed to a convolutional layer with 32 filters and a 5x5 kernel that slides through the image in a 1x1 stride. The output of this layer is then passed to the ReLU activation function, which passes the values to the pooling layer with a size of 2x2 and a stride of 2. This down-samples the input by taking the maximum value of each input window. This convolution-pooling combination is repeated three times and the number of filters in each convolution layer is 32, 64 and 64, respectively. After this, the data passes through a layer that flattens the multi-dimensional array into a single-dimensional array, which is then fed into the final classification layer with 6 nodes (one for each class). The diagram of this architecture is presented in Figure 6.8.

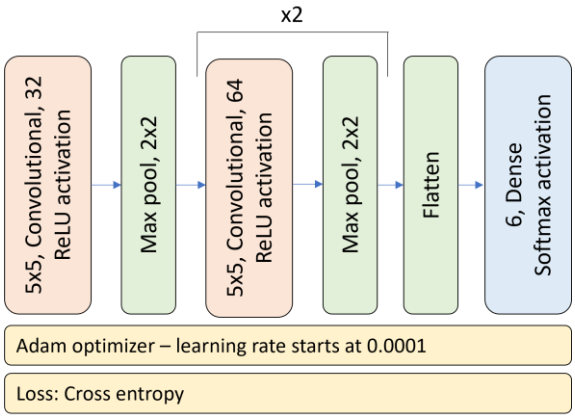


Figure 6.8 – CNN with max pooling without dropout layers architecture.

CNN with max pooling and dropout layers

The difference between this model and the previous one is the addition of dropout layers, which makes it follow the same convolution-pooling-dropout combination as the first CNN developed.

With this modification, this architecture starts with a Rescaling layer that normalizes every pixel in the image, converting the [0,255] range to a [0,1] range. This is then passed to a convolutional layer with 32 filters and a 5x5 kernel that slides through the image in a 1x1 stride. The output of this layer is then passed to the ReLU activation function, which passes the values to the pooling layer with a size of 2x2 and a stride of 2. This down-samples the input by taking the maximum value of each input window. A dropout layer with a 0.25 rate is then added, dropping 25% of the connections. This convolution-pooling-

dropout combination is repeated three times and the number of filters in each convolution layer is 32, 64 and 64, respectively. The data then passes through a layer that flattens the multi-dimensional array into a single-dimensional array, which is then fed into the final classification layer with 6 nodes (one for each class). The diagram of this architecture is shown in Figure 6.9.

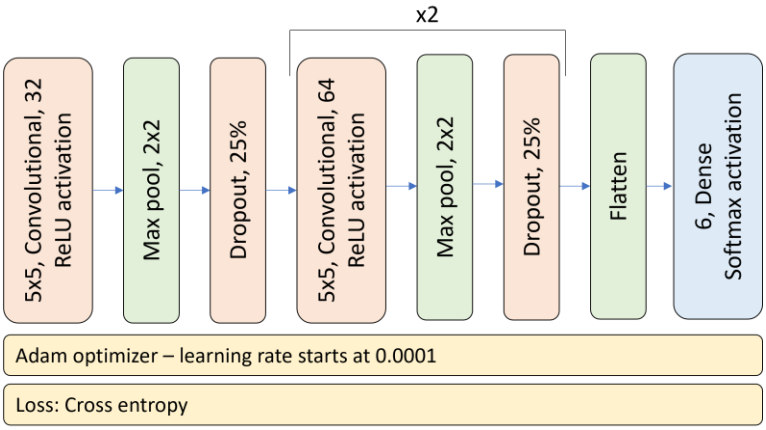


Figure 6.9 – CNN with max pooling and dropout layers architecture.

VGG-16

The fourth model developed was a VGG-16 network developed by Simonyan et al. [84]. It is publicly available and has been used as a reference architecture in many studies, such as that of Lee et al. [16], and it will also be used as a reference in this dissertation. It was pre-trained on the ImageNet dataset, which contains more than 15 million images belonging to about 22 thousand categories, but it can be adapted to different problems by maintaining the convolutional layers and their weights and removing the final dense layers used for classification. The size of the images taken by LM are 640x240, however due to the model being pre-trained on 224x224 images, the LM images were resized to match the same size.

Figure 6.10 shows the architecture used, starting with two convolutional layers, each with 64 filters, a 3x3 kernel, a stride of 1, and a ReLU activation function. The output of this layer is then passed to a pooling layer with a size of 3x3 and a stride of 2, which down-samples the input by taking the maximum value of each input window. This combination of layers is repeated one more time, changing only the number of filters in the convolution layers from 64 to 128. Then, three convolutional layers are added with the same specification except that the number of filters is the increased to 256. This output is passed to a max pooling layer with the same specifications and purpose as the previous ones. Then, three convolutional layers are added with the same specifications except that the number of filters is increased to 512. This output is passed to a max pooling layers with the same specifications as previous ones. The combination of these 3 previous convolutional and pooling layers is repeated one more time to complete this complex architecture. After the feature extraction is performed by the pre-trained VGG-16 model, it is

possible to add four dense layers of 250, 200, 100 and 50 nodes respectively. The data then passes through a layer that flattens the multi-dimensional array into a single-dimensional array. At last, a final dense output layer is added which, through the softmax activation function, gives a probability to each of the 6 classes.

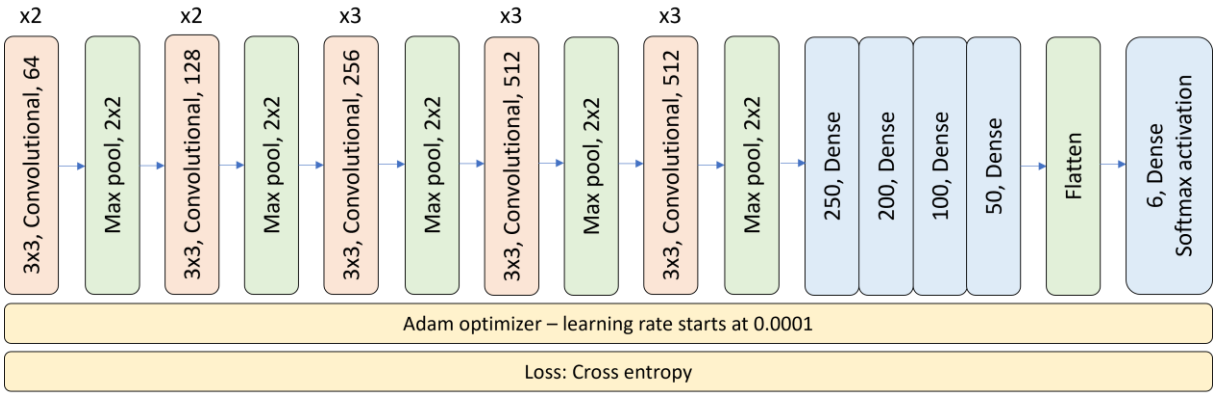


Figure 6.10 – VGG-16 architecture.

ResNet-50

The fifth model developed was a ResNet-50 network developed by He et al. [85] in "Deep Residual Learning for Image Recognition". It is also publicly available and used as a reference architecture and has the same adaptation techniques as the VGG-16 architecture. The model used was also pre-trained on the ImageNet dataset and as such, the images were also resized to 224x224 pixels.

This architecture starts with a convolution layer with 64 filters, a 7x7 kernel and a stride of 2. This is then passed to a pooling layer with a size of 3x3 and a stride of 2, which down-samples the input by taking the maximum value of each input window. Four convolution blocks are then added to the architecture. The first block has 3 sets of the following 3 convolutional layers: one with 64 filters and a 1x1 kernel, one with 64 filters and a 3x3 kernel, and one with 256 filters and a 1x1 kernel. The second block has 4 sets of the following 3 convolutional layers: one with 128 filters and a 1x1 kernel, one with 128 filters and a 3x3 kernel and, and one with 512 filters and a 1x1 kernel. The third block has 6 sets of the following 3 convolutional layers: one with 256 filters and a 1x1 kernel, one with 256 filters and a 3x3 kernel, and one with 1024 filters and a 1x1 kernel. The last block has 3 sets of the following 3 convolution layers: one with 512 filters and a 1x1 kernel, one with 512 filters and a 3x3 kernel, and one with 2048 filters and a 1x1 kernel. After the feature extraction is done by these layers in the pre-trained ResNet50 model, the output is passed to four dense layers of 250, 200, 100 and 50 nodes respectively. The data then passes through a layer that flattens the multi-dimensional array into a single-dimensional array. At

last, a final dense output layer is added which, through the softmax activation function, gives a probability to each of the 6 classes. This architecture can be seen on Figure 6.11.

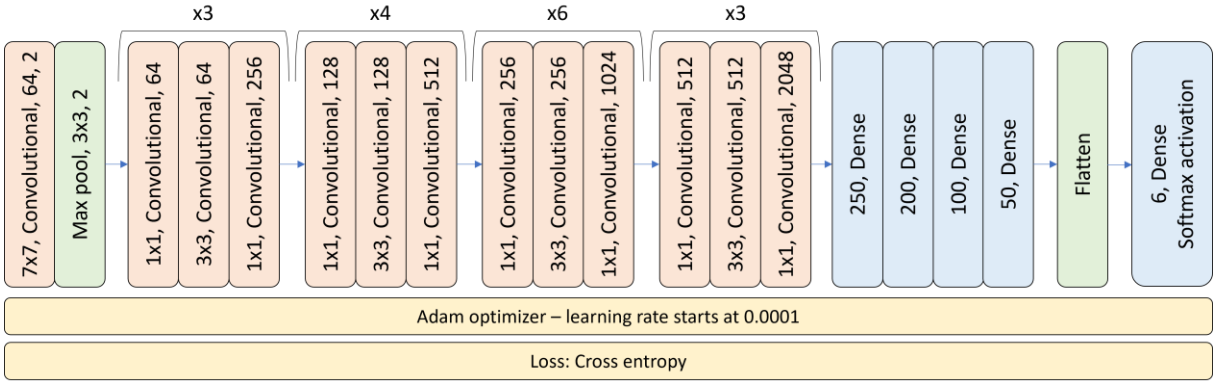


Figure 6.11 – ResNet-50 architecture.

This architecture is much more complex than the VGG-16 and proposes an innovative solution to the vanishing gradient problem that occurs when architectures become very deep. The skip connection explored in Kaiming's paper [85] is an approach that allows the model to have nested blocks without the model suffering from the vanishing gradient problem by adding a shortcut path that allows the model to bypass each set of convolutional layers. The decision to skip is evaluated by calculating the difference between the desired output and the input, called the residual. If the residual is pushed to 0, the input data passes through the set of convolutions unchanged.

6.3 Results and Discussion

This subsection presents the results obtained from the training process for each of the architectures explored in the previous subsection. First, the approach taken in the training phase is explained in detail, such as the split used on the dataset, the number of epochs, and other training parameters, followed by the evaluation metrics used to assess the performance of the models and the reasoning behind them. The results of the models using the feature dataset are presented and discussed, followed by the results of the models using the image dataset. After eliminating the worst-performing models, some data augmentation techniques were used, and the architecture of the remaining models was tweaked to improve the performance of the models. Finally, some conclusions are made.

6.3.1 Training approach

Before feeding the model, the dataset was split into three separate sets (training, validation, and test) to ensure that the results obtained were reliable. The training set contains the data used to train the

model. The validation set is used to provide an unbiased assessment of the model while it tunes the hyperparameters. The test set contains data that will not be used to train the model, so it will help understand if the model was able to generalize the information from the training set so it can accurately predict data that it has never seen before.

Since the objective of this model is to recognize gestures from any person, it is important to isolate each person into a single set of data. For the model to be able to generalize correctly, the dataset will be split based in subjects, such that the model is able to be verified in subjects that were not used for training and thus avoiding bias. Following these considerations, the test set consisted of 20% of the 21 volunteers, which resulted in 4.2 persons, rounded up to 4 to preserve volunteer isolation. The remaining 80% of the volunteers will be further split into 15% for the validation set, resulting in 2.55, so 3 volunteers will be used. The remaining 85% translates into 14 volunteers to be used in the training set.

As the description of the dataset shows, there are twice as many male subjects as female subjects. This can lead to misleading results if the test set does not contain any female volunteers because it is not possible to correctly assess if the model is also capable of generalizing for female users. To have a representative sample, from the four volunteers used in the test set, two are female and two are male.

Cross-validation was used to prevent the model from overfitting on a specific dataset. This technique also enabled to train various models on the entire dataset.

To have more confidence in the performance of the models, 9 folds were used, respecting all the points already explained.

6.3.2 Evaluation metrics

After the training process, it is important to evaluate the performance of a model. Several metrics are available to developers for this purpose. As evidenced by the review made on Chapter 3, the most used metric for automatic gesture recognition tools is accuracy, and it is generally a good metric for assessing the general ability of the model to correctly predict data if the dataset is balanced, which is the case of this dissertation. However, some other metrics and tools provide a deeper understanding of the model's behaviour. The loss function is very important because it helps to understand how well the model is behaving. In a 6-class classification problem like this one, high accuracy but high loss can mean that the model is predicting correctly but with low confidence, and it is important to have this knowledge. Across every model studied, categorical cross entropy will be the loss function used. The confusion matrix is a NxN matrix, where N is the number of classes, which compares the actual target values with those predicted by the model. This gives a wider view of the capability of the model to predict each class,

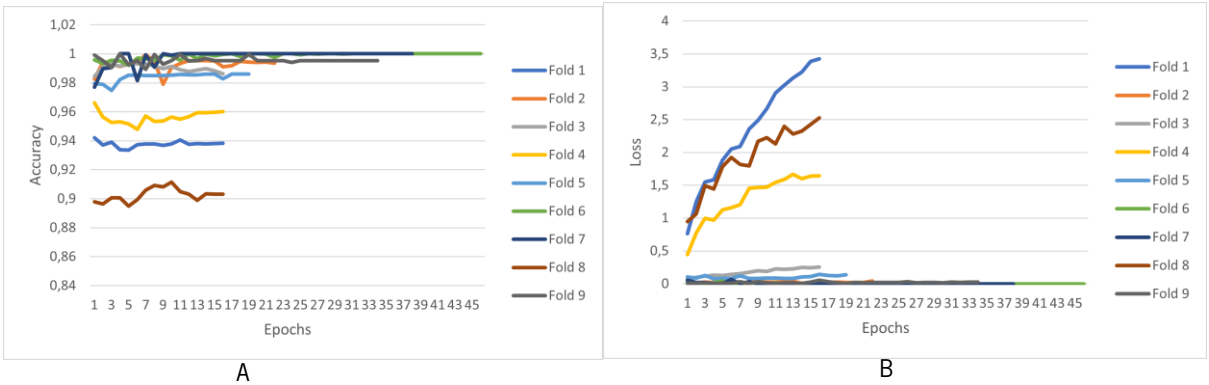
allowing the developer to understand which classes the model is misclassifying. Since the ultimate goal is to use one of these models in a real-time application, the model's time inference is also very important, so it was also monitored as an evaluation metric.

6.3.3 Feature-based data

This subsection explores the results obtained by each of the architectures defined in subsection 6.3.1 and, for each model, provides a comparison with the results of the same architectures from the reviewed studies mentioned above. All these models were fitted to a maximum of 50 epochs, with a batch size of 25, and had 3 callbacks that ran at the end of each epoch. One evaluates whether the validation accuracy has increased compared to the previous epoch and, if so, saves the state weights of the model to a file as a checkpoint. The second callback reduces the learning rate of the model by a factor of 0.5 if the validation loss does not improve over 5 epochs. The final callback stops the fitting of the model early if the validation loss has not improved over the last 15 epochs.

NN with linear activation

The NN with linear activation achieved 99.39% of accuracy on the training data, 97.76% of accuracy on the validation data, and 92.72% of accuracy on the test data. These results show that the model can successfully generalize the training information to accurately predict unseen data. With this performance, this model becomes a possible candidate to be used in the final solution. Figure 6.12 presents the validation accuracy and loss progress over the epochs for each of the 9 defined folds.



A B

Figure 6.12 – NN with linear activation model behaviour on the validation data over the epochs: (A) on the accuracy, and (B) on the loss.

Looking at Figure 6.12, on the validation accuracy, the difference between folds is very slim, which also supports the success of the model. As for the validation loss, it increases along the epochs in some folds, which indicates that the model is deteriorating. However, this growth is contained by the implemented early stop callbacks.

The confusion matrix for this model is illustrated in Figure 6.13 and shows that the model was able to predict most of the classes without any problems. However, there was a noticeable confusion in the model between the zoom out gesture and the move gesture. This may be due to the fact that the zoom out gesture, when made at certain angles, can be too similar to the move gesture. There was also some confusion between the move gesture and the click gesture, and between the mouse control gesture and rotate gesture but, overall, the model shows good ability to make correct predictions.

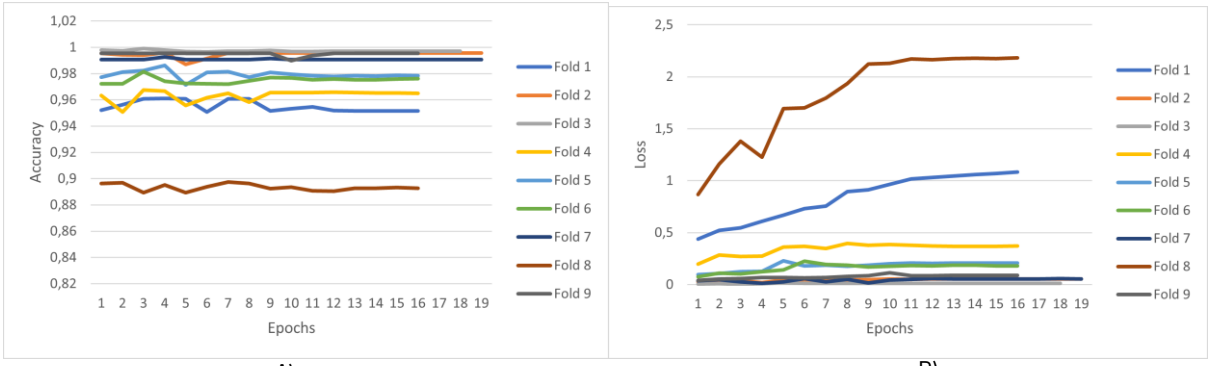
Figure 6.13 – Confusion matrix for the linear activation NN.

After converting the TensorFlow model into an optimized version using ONNX runtime, the model's inference time becomes negligible, taking an average of 0.000263 seconds. This allows the model to be used for a real-time forecasting application. These results show that this architecture in this dataset can achieve good performance.

This architecture, based on Hu et al. [57], uses a simple NN architecture. In Hu et al. [57], the dataset consists of 9124 samples for training and 1938 samples for testing collected from 7 participants, and only uses 15 features out of the many more provided by the LM camera. The model was trained with four different batch sizes: 25, 50, 75 and 100, and their respective testing accuracies were 98.55%, 97.32%, 98.19% and 98.09%. Despite the high accuracies, Hu et al. [57] do not mention if they isolated the participants from the training data to the test data, so the high performance of the model may be due to the fact that it is testing with characteristics of people it already knows. Consequently, it is not possible to assess whether the model presented will behave in the same way in a real situation when trying to predict new people. Thus, it is difficult to make a direct comparison between the models, as they are presumably not under the same conditions.

NN with ReLU activation

This NN was trained with ReLU activation in the four dense layers, instead of the previous NN explored that used linear activation, to explore what effect this change in the architecture would have on the results. With this change, the model achieved, across the 9 folds, an average accuracy of 99.99% on the training data, 97.52% on the validation data, and 93.82% on the test data. The change is quite small, but the overall accuracy of the test increases. Figure 6.14 shows the progression of validation accuracy and loss over the epochs.



A) B)

Figure 6.14 – NN with ReLU activation model behaviour on the validation data over the epochs: (A) on the accuracy, and (B) on the loss.

The validation accuracy and loss behaviour over the epochs is identical to that of the previous model with a linear activation function. The validation loss continues to increase slightly in some folds, but at a slower rate and stabilizes at a lower value. Despite the small changes, the slight increase in test accuracy and the slight decrease in the overall loss indicate that this model performs better than the previous one.

Figure 6.15 shows the confusion matrix of the previous model. The misclassification between zoom out and move gestures, between click and zoom in gestures, and between click and zoom in gestures, have decreased slightly, but this model has more difficulty distinguishing between the zoom out and the zoom in gestures (1.9e+02), than the previous model (38). As a result, there is no clear and straightforward choice between this model and the previous one as to which is the best.

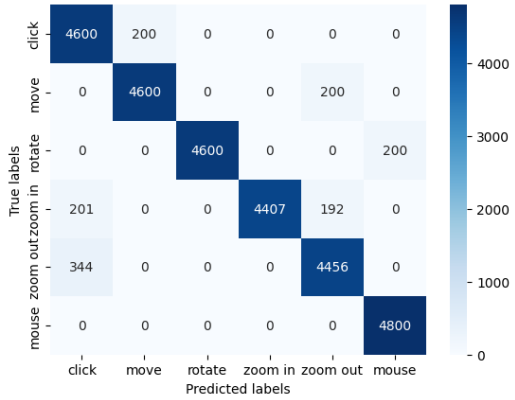


Figure 6.15 – Confusion matrix for the ReLU activation NN.

After converting the TensorFlow model to an ONNX model, the average inference time is about 0.000495 seconds, which also makes this model very viable in the context of the intended application.

SVM

Given that SVM (ML algorithm) is simpler than the previous DL models, and that the feature-based dataset was reduced from the original 262 featured to just 16, there was an opportunity to use the GridSearchCV tool from the scikit-learn module to perform a broader search for hyperparameters and thus tune the model to obtain the best possible performance. The following were tested: C parameter values of 0.5, 1 and 10; gamma parameters values of 1, 5 and 10; and linear and radial basis function kernels. After fitting, the best parameters were 0.5 for C and 1 for gamma in a linear kernel, with an accuracy of 89.18% on the train data and 96.25% accuracy on the unseen test data. Due to the nature of the implementation of this algorithm, the training and validation sets defined previously were joined together, forming the train set used on this model. The test set remained the same and achieves a better accuracy than the training data because the model can learn and generalize the training information, and correctly predict on unseen data. Figure 6.16 shows the confusion matrix for the test data.

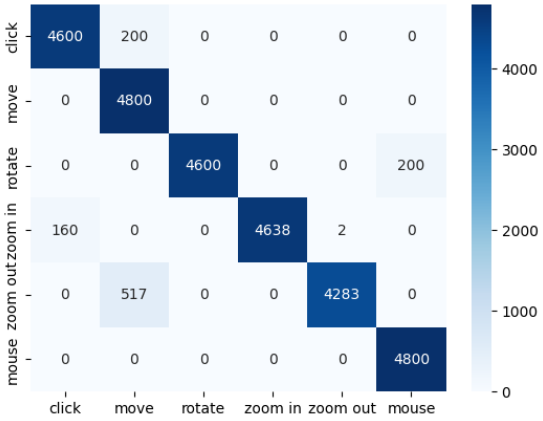


Figure 6.16 – Confusion matrix for SVM.

Looking at Figure 6.16, the model showed a good ability to recognize between most of the gestures, with only a slight confusion between the move and the zoom out gestures. This may be understandable, as the move gesture may look too similar to the zoom out gesture at some angles. The average inference time for this model is 0.007192 seconds, making it a possible candidate for use in the final solution.

Decision Tree

The second ML model was developed using a Decision Tree algorithm. For the same reasons as for the SVM model, the GridSearchCV tool was also used in the development of this model, to obtain the best hyperparameters from a predefined set. The values tested for the function measuring the quality of a split (criteria) were gini and entropy; the values for the maximum depth of the tree were 10, 15 and 20; the values for the minimum number of samples required to split an internal node (min_samples_split) were 2, 5 and 10; and the values for the minimum number of samples required to be at a leaf node (min_samples_leaf) were 1, 5 and 8. After the fitting process, the best parameters were entropy for the criteria, 20 for the maximum depth, 2 for min_samples_split, and 8 for min_samples_leaf, with 73.51% accuracy on the training data and 77.84% accuracy on the unseen test data. Compared to the previous ML algorithm, the SVM achieved much more impressive results. The confusion matrix in Figure 6.17 shows that the model can recognize several gestures but has great difficulty distinguishing the zoom in from other gestures, making it not very reliable overall.

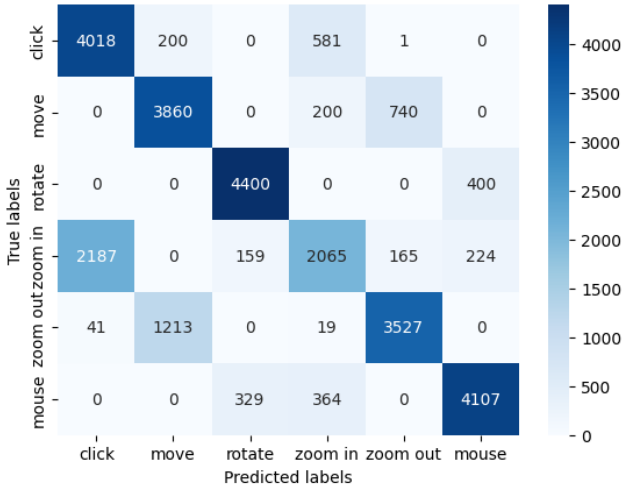


Figure 6.17 – Confusion matrix for Decision Tree.

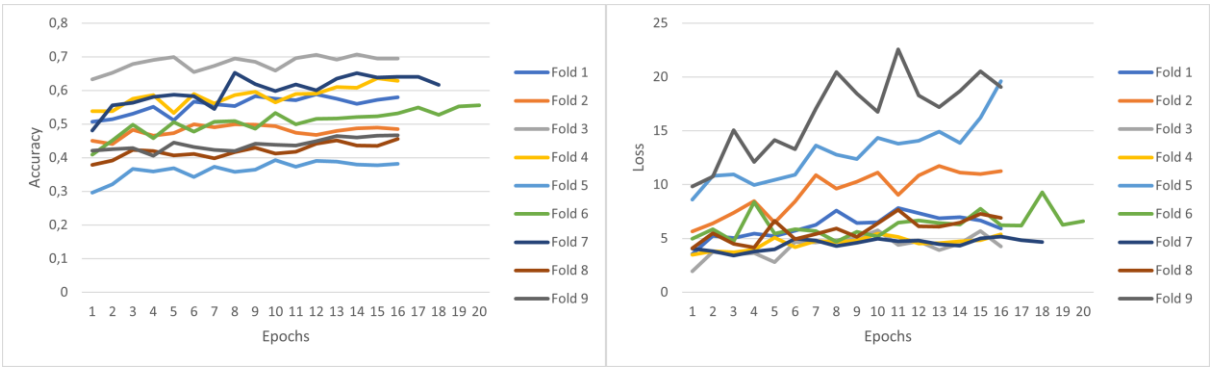
The average inference time of this model is around 0.002485 seconds, which shows that it can deal with the time sensitive problem in question. However, as seen, its performance is not very good.

6.3.4 Image-based data

This subsection examines the results obtained by each of the architectures defined in Section 6.3.2 and presents some comparisons with the results obtained in the original studies. All these models were fitted for a maximum of 20 epochs, with a batch size of 32, and 3 callbacks that ran at the end of each epoch. One of these evaluates if the validation accuracy has increased since the last epoch and, if so, saves the weights of the model state to a file as a checkpoint. The second callback reduces the model learning rate by a factor of 0.5 if the validation loss does not improve over 5 epochs. The final callback stops the model fitting early if the validation loss has not improved over the last 15 epochs.

CNN with average pooling

CNN with average pooling obtained 99.89% accuracy on the training data, but only reached 54.66% accuracy on the validation data, and on the test data the model did not exceed 53.88% accuracy. The evolution of the validation accuracy and loss across the 9 folds can be seen in Figure 6.18.



A) B)

Figure 6.18 – CNN with average pooling model behaviour on the validation data over the epochs: (A) on accuracy, and (B) on loss.

The validation accuracy between folds is very variable and barely increases over the epochs, while the loss generally increases over the epochs. These results show a clear overfitting of the training data, because it achieved very good results on training, but was unable to generalize the learned features to the rest of the data. The model has not learned the differences between the images, and as such it is unable to correctly predict on new data. As described before, this architecture has a dropout layer at the end of each convolution-pooling combination of layers, which drops 25% of the connections. This was done to avoid overfitting, but it clearly was not enough to solve the problem.

The confusion matrix presented in Figure 6.19 also shows the lack of performance of this model, as it was unable to correctly distinguish any gesture, having particularly difficulty in distinguishing between the click and any other gesture.

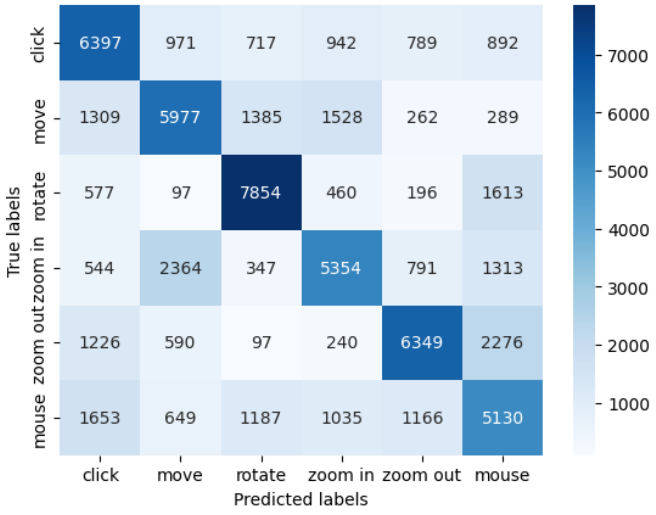


Figure 6.19 – Confusion matrix for CNN with average pooling.

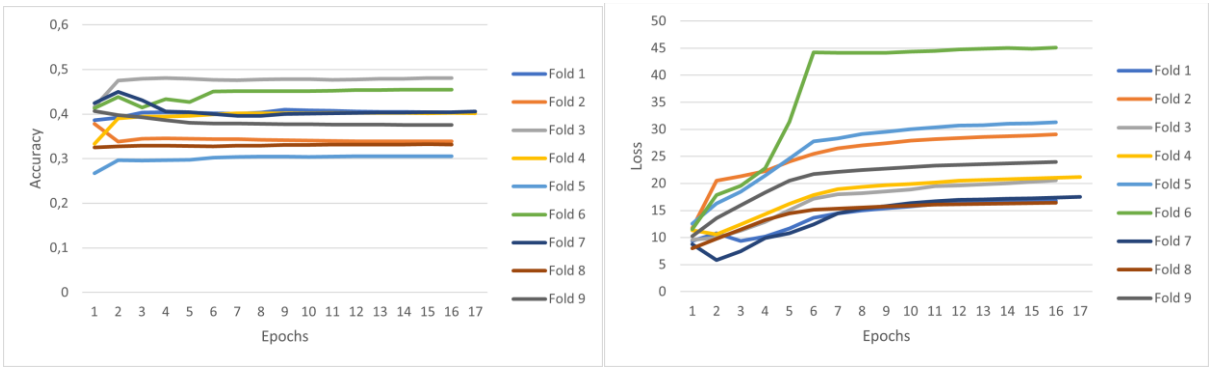
Despite the lack of performance, this model was still converted to the ONNX model to check for the inference time. The average inference time is about 0.022611 seconds, making this model suitable for the application in terms of responsiveness.

This CNN architecture was based on Enikeev et al. [55], who presented a dataset of 800 samples (4 people performing 10 gestures 20 times). In this study, the model was trained for 5 epochs on 560 samples and evaluated on the remaining 240 samples of the dataset. Enikeev et al. [55] presented results with an accuracy of 97% for the training data and 97.5% for the test data. However, this performance is not reflected in the model developed in this dissertation. This difference in results is probably due to the way the dataset was prepared. Enikeev et al. [55] do not mention the splitting technique used to separate the data into training and test sets, which is very important. The authors probably randomly split the data, without bothering to isolate each individual on only one side of the split, as was done in this dissertation. This is a problem because if this is not done, the model is evaluating data that it has already seen in the training data, so it is more likely to classify it better. In this dissertation, by separating four people for the test split, there are always four people worth of samples that the model has never seen before, so the model's ability to generalize the features for other people is better.

CNN with max pooling without dropout layers

The second model developed also has a CNN architecture. The main differences between this architecture and the previous one is the replacement of the average pooling layer by a max pooling layer, the increase in the number of filters on the convolution layers, and the absence of dropout layers. After training, the accuracy was very high at 99.68%, but once again the model failed to generalize, and the accuracy of the validation data is only 40.22%. The test data follows the same trend as expected, only

achieving an accuracy of 35.08% on unseen images. The validation accuracy and loss progression for the 9 folds can be seen in Figure 6.20.



A) B)

Figure 6.20 – CNN with max pooling without dropout layers model behaviour on the validation data over the epochs: (A) on accuracy, and (B) on loss.

The behaviour of these metrics was the same for all folds. The validation accuracy started low and did not improve over the epochs. On the other hand, the validation loss increased over the epochs, which means that the model is getting worse over time because there is an increasing discrepancy between the model predictions and the actual ground truth labels. This suggests that the model is becoming increasingly uncertain in its predictions.

The confusion matrix illustrated in Figure 6.21 corroborates the lack of performance. The model was unable to recognize between any gestures, showing even worse performance than the first CNN architecture explored.

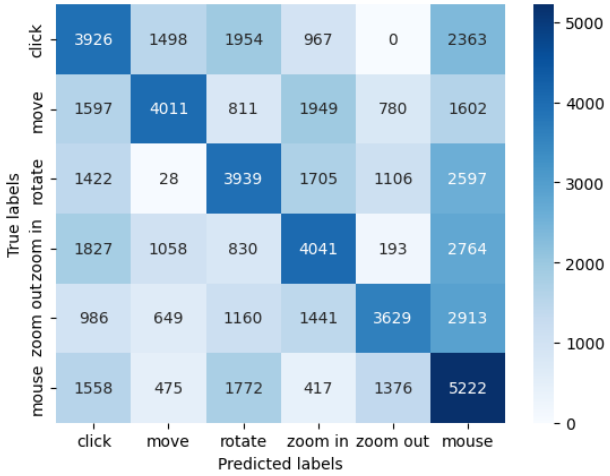


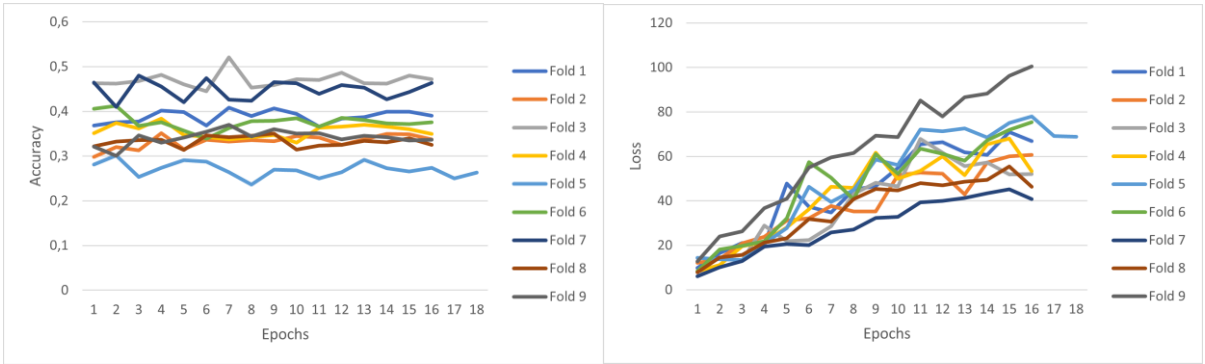
Figure 6.21 – Confusion matrix for CNN with max pooling without dropout layers.

After converting the TensorFlow model to the more efficient ONNX runtime, the average inference time is about 0.06884 seconds, making the model capable of handling the time-sensitive task despite its very poor performance.

This architecture was based on the work of Tripathy et al. [18]. Their dataset is described by 6 gestures (3 with the left hand and 3 with the right hand), each with 2000 samples, for a total of 12000 samples. In this study, the model was trained for 20 epochs with a batch size of 64 and achieved an accuracy of 98% on the training data and 99% on the validation data. As with the first CNN explored, the results of the article do not agree with the results of this dissertation. The difference between these results may be due to the same reason given for the first architecture. The authors do not specify the splitting technique or even the splitting percentage used, but they most likely did not isolate some subjects to the validation set only, so these results suffer from the same problem as the previous article.

CNN with max pooling and dropout layers

The third architecture developed also follows a CNN architecture. The main difference between this architecture and the previous one is the addition of a dropout layer that drops 25% of the connections at the end of each convolution-pooling combination, to understand if this addition in the architecture has a positive impact on the results, and the model can generalize better. With an overall training accuracy of 99.92% and a validation accuracy of 39.75%, it is safe to say that the problem cannot be solved by simply adding dropout layers but lies in the overall architecture or in the dataset. The test data was also similar to the model without the dropout layers with an accuracy of 34.64%. The progression of validation accuracy and loss can be analysed in Figure 6.22.



A) B)

Figure 6.22 – CNN with max pooling and dropout layers model behaviour on the validation data over the epochs: (A) on accuracy, and (B) on loss.

The validation accuracy fluctuates more than in the previous architecture, but it still does not learn over the training process. The validation loss explodes, making the model worse at each epoch, and indicating that the addition of the dropout layers only makes the model worse because it has less information to make a confident prediction.

Figure 6.23 presents the confusion matrix for CNN with max pooling and dropout layer model. The low accuracies resulted in a very poor matrix, showing a clear inability of the model to distinguish between any gestures.

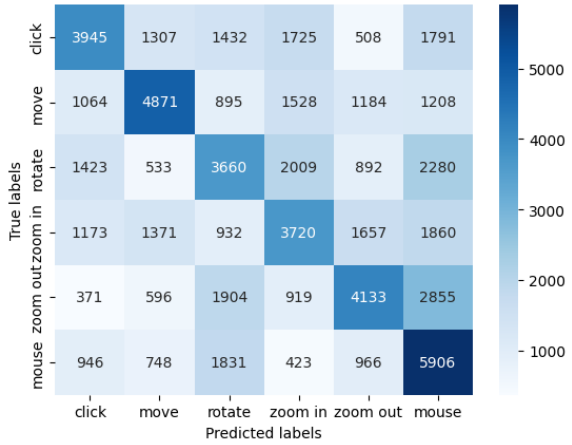
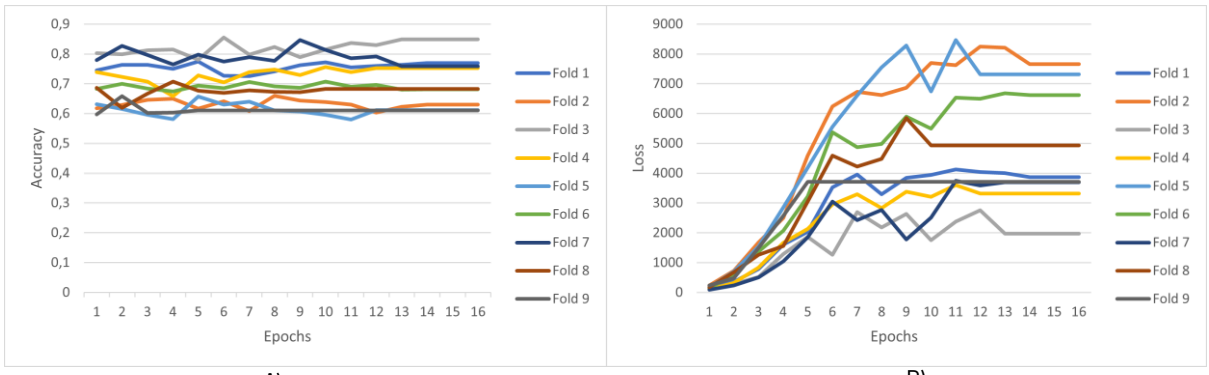


Figure 6.23 – Confusion matrix for CNN with max pooling and dropout layers.

Despite the low performance of this model, since it has approximately the same complexity as the previous two architectures, the inference time remains low after converting the model to the ONNX runtime, with an average of 0.06451 seconds.

VGG-16

The fourth architecture tested was an adapted VGG-16, which has a good reputation for extracting the features from images and achieving good classification results. The training accuracy was 99.88% and the validation accuracy was 73.58%. This model achieved a much better performance compared to the previous three CNNs, but it is still not good as the test data only achieved an accuracy of 65.29%. In Figure 6.24, it is possible to analyse the progress of the validation accuracy and the loss on the 9 folds.



A) B)

Figure 6.24 – VGG-16 model behaviour on the validation data over the epochs: (A) on accuracy, and (B) on loss.

The validation accuracy started high but remained at the same value over time. Looking at Figure 6.24B, the model gets worse over the epochs, as the loss explodes to very high numbers, indicating a poor performance of the model.

The confusion matrix in Figure 6.25 shows that, despite the relatively high accuracies, the model had great difficulty in distinguishing most of the gestures. This shows that the model needed some work to be viable.

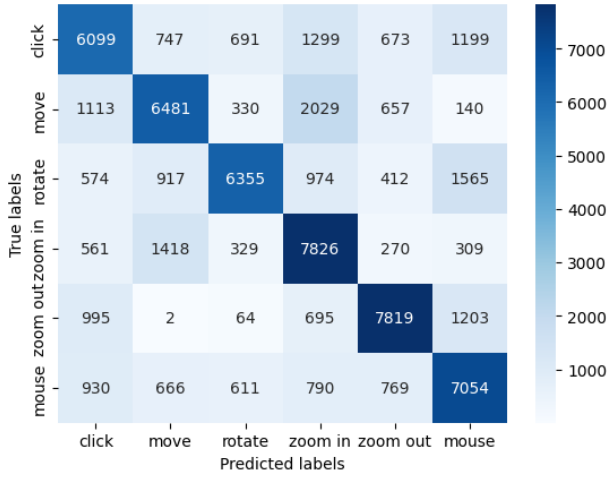


Figure 6.25 – Confusion matrix for VGG-16.

After making the model more efficient using the ONNX runtime, the average inference time is 0.24264 seconds. Considering the much greater complexity of this architecture, this higher time is understandable, but makes this model unusable in the context of the required time-sensitive application.

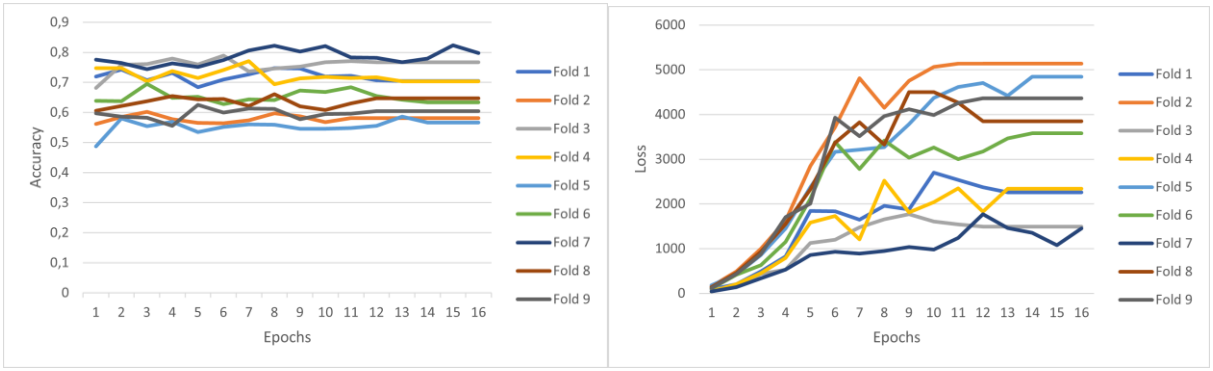
This architecture was based on the work of Lee et al. [16], and for the first time there is some correlation between the results. Their raw dataset consists of 1003 samples obtained from 10 subjects performing 5 different gestures, but they used geometric techniques (rotation, resizing, and scaling) and photographic techniques (translation and illumination) to augment the original dataset by a factor of 5.

The model "was trained on 200 epochs and 20 batch sizes" and achieved a classification accuracy of 76.4%. This correlation is probably due to a similar approach in how the dataset is prepared for the training step. The authors used 80% of the dataset for training and the remaining 20% for validation, but most importantly they use a leave-one-out cross-validation per person approach. In this manner, the model does not learn on the people that will later be used to evaluate the model, giving a better understanding of whether the model is correctly generalizing the information.

In addition to this correlation, it is also possible to see that this architecture performed better than the previous 3 CNNs tested, and since VGG-16 is known to have a very good image recognition capability, this result was expected. This higher performance is due to two reasons: firstly, it has more combinations of convolution-pooling layers, and secondly, each convolutional layer has more filters, so it is expected to extract more features from the images and consequently, achieve higher performance than previous models. This increase in performance is good, but still not enough to be used reliably.

ResNet-50

The final architecture being tested was a ResNet-50, which is another well-established CNN in the DL field. The implementation of this architecture was not based on any literature study, so its results will not be directly compared to another study. However, this architecture was also considered as a first approach to understand its results. The training accuracy obtained was 99.91% with a validation accuracy of 69.99%. The model performed worse on the unseen test data, with an accuracy of only 60.44%, making this model slightly worse than VGG-16. A more detailed view of the progression of the validation accuracy and loss can be seen in Figure 6.26.



A) B)

Figure 6.26 – ResNet-50 model behaviour on the validation data over the epochs: (A) on accuracy, and (B) on loss.

These graphics show a similar behaviour to the VGG-16 model. The validation accuracy starts relatively high but does not improve over the epochs. Meanwhile, the validation loss starts low and explodes through the epochs on each fold, making the model worse at each epoch.

The confusion matrix presented in Figure 6.27 corroborates the lack of performance of the model. It is difficult to distinguish between all the gestures, making it unusable.

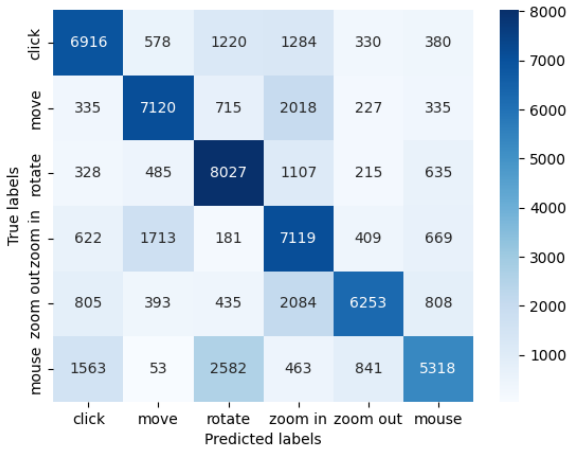


Figure 6.27 – Confusion matrix for ResNet-50.

Considering that the ResNet-50 architecture is also very complex, the inference time was expected to be similar to that of the VGG-16. However, after testing, it averaged only about 0.08516 seconds, making the model usable in terms of prediction time.

This result shows that this architecture also had a better capability to extract features from the images than the first 3 CNNs explored. This was expected because, like the VGG-16 architecture, ResNet-50 also has more convolution-pooling layers, but this is still not high enough to be applied in the real world. This difference in performance may be due to the over-complexity of the ResNet-50 architecture compared to the VGG-16, which makes it unable to generalize the information.

6.3.5 Improvement of the models

Since there is no single architecture among the image-based data models that has achieved good enough results to be applied in a real-world application, the five architectures discussed in the previous section will serve as basis for improving the performance of the models. Rather than wasting time on architectures that gave very poor results from the start, it is better to focus on improving those that gave some better results and have the potential to improve even more. Therefore, the second and third architectures developed (CNN with max pooling without dropout layers and CNN with max pooling and dropout layers) were dropped. Both were based on the CNN from the same literature study [18] , and

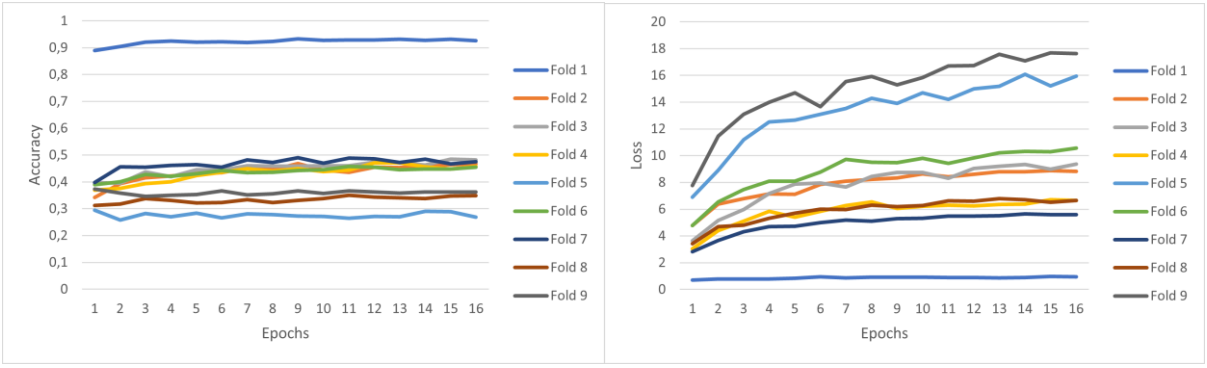
they did not manage to exceed 41% accuracy on either validation data or test data, so the expectations of this architecture thriving are low. All the remaining architectures were considered for improvements.

There are various strategies to enhance the performance of the developed models. One such technique is data augmentation, which increases the diversity of the training data by applying a series of random transformations. This technique can have a very strong impact on the accuracy of a given model by giving the model a broader set of scenarios to predict, making the model more robust when faced with new information. TensorFlow offers a broad set of transformations, including resizing, rescaling, flipping, and rotating the images. Another strategy to enhance the performance of the developed models is by analysing the results of the model and adapting the architecture and/or the hyperparameters accordingly. Both these strategies were used to enhance the model's effectiveness.

At first, the first strategy (image augmentation) was employed to all models explored below. The rotation transformation rotated the image randomly from the range [-20%, 20%], and the translation transformation shifted the image vertically randomly on the range from [-20%, 20%], and horizontally randomly on the range from [-40%, 40%]. The second strategy (adapting the architecture) was employed in conjunction with the first strategy for some of the models.

CNN with average pooling

The image augmentation technique reduced the overall validation and testing accuracies of the CNN architecture, which already had a very low performance. The validation accuracy and loss of the model over the epochs for each fold can be seen in Figure 6.28.



A) B)

Figure 6.28 –CNN with average pooling model behaviour, trained on the augmented dataset, on the validation data over the epochs: (A) on accuracy, and (B) on loss.

This model was not able to learn over the epochs and, not considering the 90% accuracy achieved in the first fold, the model is between 30% and 50% accuracy depending on the fold. The loss started

relatively low but increased over the 16 epochs, showing the opposite of the desired behaviour. The lack of learning ability of the models may be due to the architecture not being able to deal with the increased complexity of adding image augmentation.

The confusion matrix in Figure 6.29 shows that the model was unable to recognize between any of the gesture, corroborating the very poor performance exhibited by the validation accuracy.

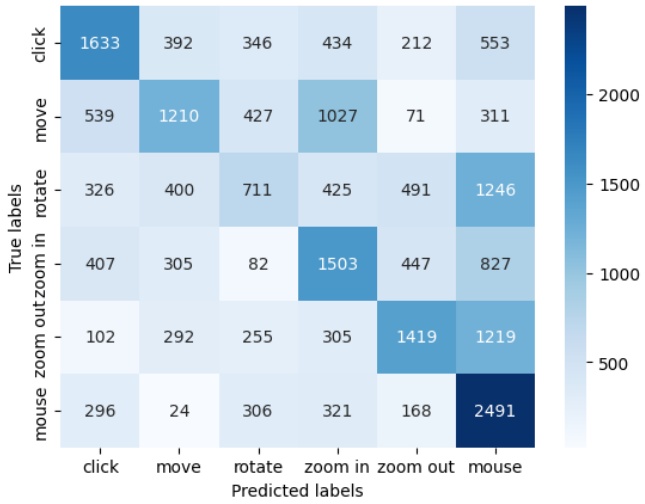
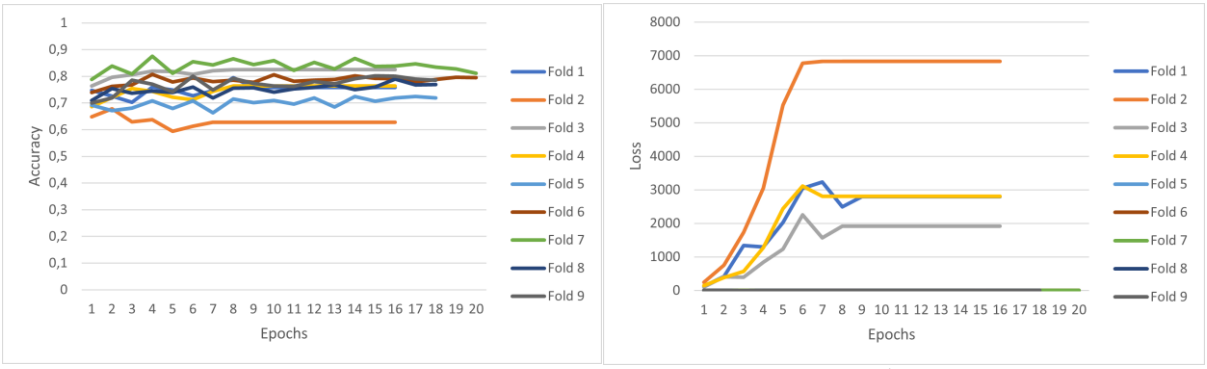


Figure 6.29 – Confusion matrix for augmented CNN with average pooling, trained on the augmented dataset.

As this is still a simple CNN architecture, after converting the model to the ONNX runtime, the inference time averages 0.043996 seconds, making the model viable in terms of prediction time.

VGG-16

On the VGG-16 architecture, the validation and test accuracy barely changed with 73.4% and 65.74%, respectively. Figure 6.30 show the validation accuracy and loss of the model at each epoch.



A) B)

Figure 6.30 – VGG-16 model behaviour, trained on the augmented dataset, on the validation data over the epochs: (A) on accuracy, and (B) on loss.

The validation accuracy was relatively high, but it did not change over the epochs, showing that the model was not able to learn with each new epoch. Despite this relatively high accuracy, the validation loss started very low and then exploded, indicating that the model was getting worse over the epochs, meaning that although the model accuracy remained relatively the same, it was making these predictions with much less confidence. There was also a high variability of the loss behaviour between the folds that made the model unreliable and did not allow to understand whether the image augmentation technique had the potential to help the model's performance. There are several reasons for the exploding validation loss, but it is likely to be due to a low number of nodes on the dense layers used for classification.

Figure 6.31 illustrates the confusion matrix of this model on the unseen test data.

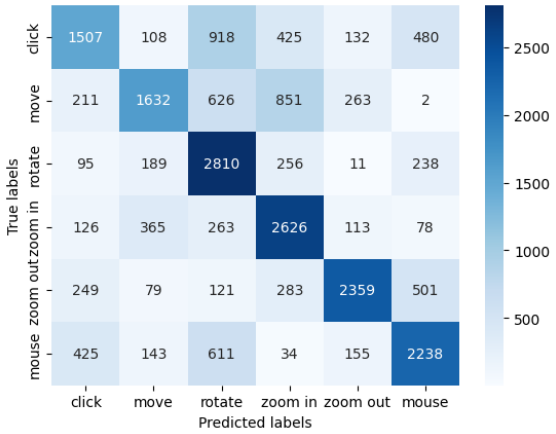
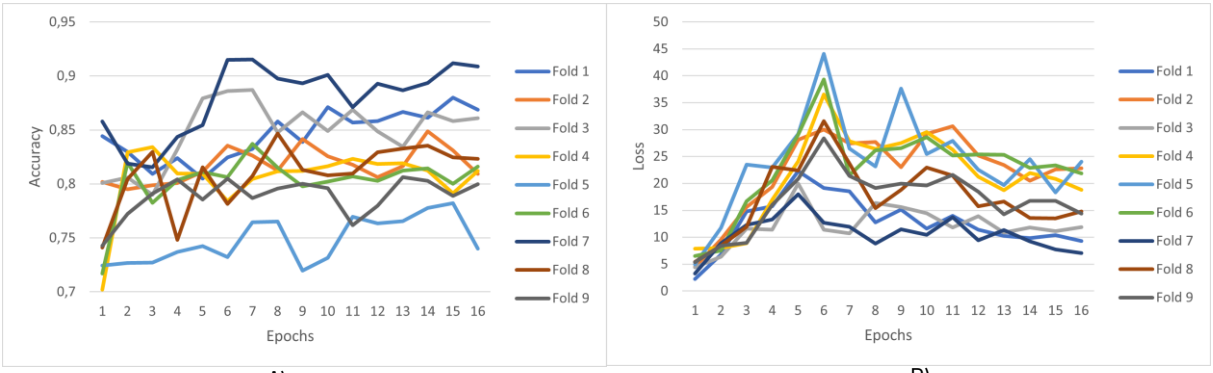


Figure 6.31 – Confusion matrix for VGG-16, trained on the augmented dataset.

The analysis of Figure 6.31 corroborates the lack of performance of this model, which has a very clear difficulty in distinguishing between each class, but especially between the click gesture and any other gesture.

The VGG-16 architecture is much more complex and, as it was shown in the previous section, the inference time of this model after ONNX conversion averages at 0.20689 seconds, rendering the model unusable in the final application.

Given the complexity of the VGG-16 architecture and the large number of features it extracts, the 250 nodes are likely to be insufficient for the model to capture all the diverse and intricate patterns in the data. The last four dense layers of the architecture were then replaced by two layers of 2048 nodes each. This change had a major impact on performance, both in terms of accuracy and, perhaps more importantly, in terms of model losses. With this change, the validation accuracy increased to 84.85% and the test accuracy increased to 78.16%. The evolution of validation accuracy is shown in Figure 6.32.



A) B)

Figure 6.32 – 2-layers VGG-16 model behaviour, trained on the augmented dataset, on the validation data over the epochs: (A) on accuracy, and (B) on loss.

The validation accuracy now fluctuates more but overall, it increases over the epochs and the loss now still increases in the early epochs to a peak of 44 in the Fold 5, but then slowly decreases to much lower values than with the previous architecture.

This change is also quite noticeable in the confusion matrix shown in Figure 6.33. In the confusion matrix of the VGG-16 with four layers of classification, it is possible to see an overall mix of misclassifications between all classes, but with the change to two layers with more nodes, it is possible to see that the model is now able to correctly identify between more gestures. However, there is a clear problem in identifying the click and move gestures.

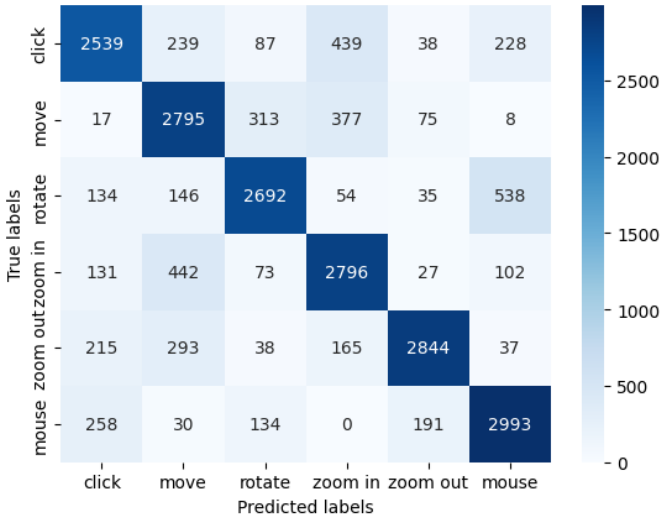
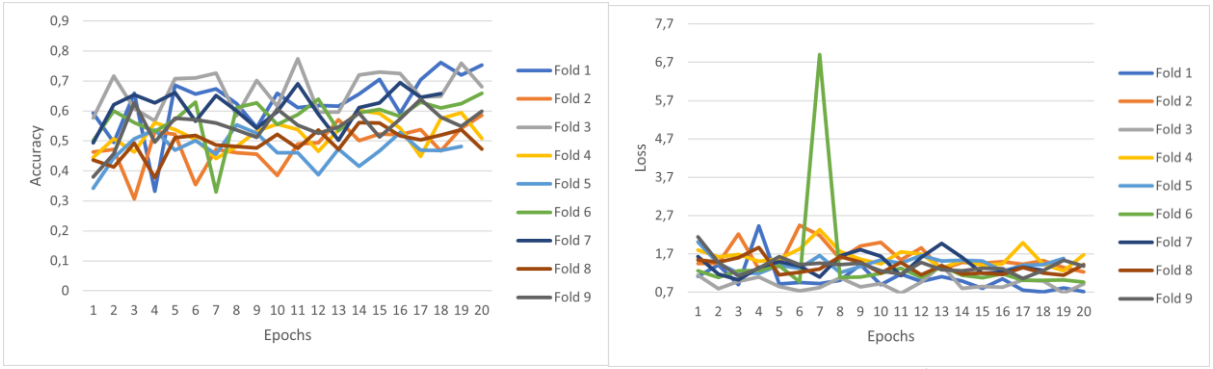


Figure 6.33 – Confusion matrix for 2-layer VGG-16, trained on the augmented dataset.

Unfortunately, after converting the TensorFlow model to the more optimized ONNX runtime, the model's inference time has an average of 0.24544 seconds, making this model unusable in the context of the system under development, where an immediate prediction time is required.

ResNet-50

The last architecture being tested was the ResNet-50. The image augmentation technique described above was applied to this architecture, as well as the replacement of the four dense layers used for classification with two dense layers with more nodes, such as the VGG-16. This change reduces the overall performance of the model, resulting in a training accuracy of 74.04%, a validation accuracy of 64.81%, and a testing accuracy of 48.68%. The evolution of the validation accuracy and loss over the epochs can be seen in Figure 6.34.



A) B)

Figure 6.34 – 2-layers ResNet-50 model behaviour (trained on augmented dataset) on the validation data over the epochs. (A) on the accuracy. (B) on the loss.

The validation accuracy is quite volatile, going up and down over the epochs but overall, the model is learning over time. The validation loss is also volatile, but it maintains the same low values over the epochs and shows a significant improvement over the previous ResNet50 architecture tested, which showed a growing loss trend to very large values.

Despite this model exhibiting a better behaviour than the first ResNet50 architecture tested, it still does not have a good enough performance to be used, as shown by the low accuracy on the test data and corroborated by the confusion matrix in Figure 6.35.

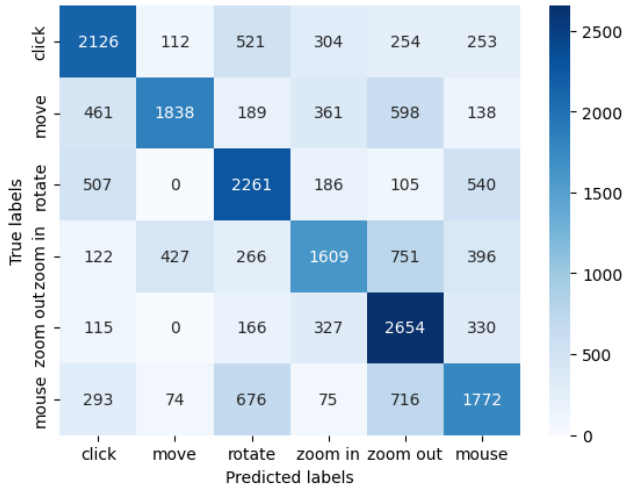


Figure 6.35 – Confusion matrix for 2-layers ResNet-50, trained on augmented dataset.

The model has a very hard time distinguishing between most of the classes, and because of the increase in the total number of nodes, the model's inference time also increases to an average of 0.12214 seconds, making it unfeasible to use in the final application, even if it did get better accuracy.

Considering all the models developed for both the feature-based and the image-based datasets, the models developed with the hand features have the edge over the image counterpart. They achieved better accuracies (feature-based: 96.25% on SVM, 93.82% on NN with ReLU activation; image-based: 78.16% on VGG-16 with data augmentation, 65.29% on VGG-16 without data augmentation) and, due to their simpler architectures, they also had very low inference times (feature-based: 0.000263 seconds on CNN with linear activation, 0.000495 seconds on CNN with ReLU activation; image-based: 0.022611 seconds on CNN with average pooling, 0.24264 seconds on VGG-16). Of the feature-based models, the best two were the SVM and the NN with ReLU activation. They only have a 2.43% difference in accuracy on the test data (SVM: 96.25%, NN: 93.82%), but on the confusion matrices this difference is noticeable, giving the SVM model the edge. The SVM had a higher inference time compared to the NN model (SVM: 0.007192 seconds, NN: 0.000495 seconds), but it is already so low that this difference is negligible. These results led to the decision to use the SVM model in the final application. The result of the decided model also fulfils the KPI defined in the objective 4 of this dissertation that stipulates the accuracy on the test data should be above 90% and inference time below 0.1 seconds. The accuracy was based on the results seen on the literature review and the inference time was based on the real time nature of the work. In Table 6.2, it is possible to observe the training, validation, and testing accuracy, and inference time (in seconds) for every model studied on this dissertation.

Table 6.2 – Training, validation, and testing accuracy, and inference time (in seconds) for every model studied on this dissertation

Models developed	Training accuracy	Validation accuracy	Testing accuracy	Inference time (seconds)
NN with Linear activation	0.994	0.9775	0.9272	0.000263
NN with ReLU activation	0.999	0.9751	0.9382	0.000495
SVM	0.8918	-	0.9625	0.007192
Decision Tree	0.7351	-	0.7784	0.002485
CNN with average pooling	0.8309	0.3766	0.3542	0.022611
CNN with max pooling without dropout layers	0.8540	0.7507	0.6976	0.06884
CNN with max pooling with dropout layers	0.8668	0.6960	0.6222	0.06451
VGG-16	0.8851	0.8728	0.8239	0.24264
ResNet-50	0.8887	0.8683	0.8324	0.08516
CNN with average pooling on augmented dataset	0.9414	0.4802	0.3998	0.043996
4-layers VGG-16 on augmented dataset	0.9623	0.8485	0.7816	0.20689
2-layers VGG-16 on augmented dataset	0.9779	0.7843	0.6902	0.24544
2-layers ResNet-50 on augmented dataset	0.7403	0.6480	0.4868	0.12214

There is still some future work that can be done to enhance the image-based models, such as experimenting a hybrid CNN-SVM architecture that extracts the image features on the CNN model and then passes them to the SVM for classification, which showed good results in Ikram et al. [58], or using the reference and publicly available EfficientNet architecture, which showed very good accuracies on the same ImageNet dataset on which the VGG-16 and Resnet-50 architectures were trained, while having a simpler architecture.

6.4 Creation of the automatic hand gesture recognition tool

After selecting the best model (SVM model – feature-based), the final tool for automatically recognizing hand gestures was developed. At the time of the development of this tool, the latest version of LM services (v5.7.2) was used, which only supported C++ development, but the AI models were developed using TensorFlow and Scikit-learn in a Python environment. Therefore, the approach was to split the tool into two different components with different purposes and to establish communication between them via sockets.

The first component of this tool was developed in C++, and it is responsible for establishing the communication with the LM services to obtain the feature information. This program starts by opening the connection to the LM device with the required properties for feature feedback. It then opens a TCP server connection to allow connectivity with the second component of this tool. The program now enters in a loop where it sends the latest available frame to the second component when it has completed its prediction.

The second component of this tool was developed in Python, and it is responsible for predicting the gesture and performing the corresponding action on the screen. To achieve this, the developed Python component imports the necessary libraries and prepares the environment for execution. In this preparation phase, the program loads up the SVM model that was previously developed and trained, loads the file that contains the scaling parameters of the features so that it can scale the real-time information before it is inferred by the model, and establishes initial variables such as the list of classes available to predict. After this first phase, the program tries to establish a TCP connection as a client to the TCP server on the first component. Once connected, the program enters a loop in which it constantly asks the server if it has a new frame to predict. If this is the case, the program reads the new frame, extracts the palm position of the hand to know where to move the cursor, scales the new information using the scaling parameters obtained before, gives the scaled information to the model to obtain a predicted class, and performs the corresponding action on the screen using the PyAutoGUI API.

Mouse control is achieved by linear interpolation between the available space above the LM camera and the available screen space. This means that if the detected hand is directly above and in the centre of the LM camera, the cursor will be in the centre of the screen. The cursor will move up and down on the screen if the hand moves in the z-axis of the camera and left and right if the hand moves in the x-axis of the camera. A mouse control that calculated the difference between the current position of the hand and the last position of the hand and changed the position of the cursor accordingly was also developed. However, since it was very difficult to balance the natural tremor of the hands with the sensitivity of the cursor, this solution was considered too imprecise and hard to control.

6.5 Conclusions

The AI models need a very large amount of information to learn patterns and understand what distinguishes one class from another. To achieve this, a protocol was built to collect 6 different hand gestures from 21 volunteers, with and without surgical gloves. At the end of this protocol, the creation of a feature-based dataset and an image-based dataset was achieved.

A good quality of the two datasets was ensured by removing invalid trials that happened during the creation process of the datasets. The visualisation on the number of frames in each class, on each of the datasets allowed to ensure that the two datasets are balanced.

Due to the wide variety of AI architectures and the need to test two different types of datasets, many models were developed to find the one that could give the best performance. The models trained on the feature-based dataset not only achieve significantly higher accuracies but also have faster inference times when compared to the models trained on the image-based dataset. These results are in line with the literature presented in Chapter 3, where most articles use models trained with a feature-based dataset, probably because they are generally better than the models trained with an image-based dataset. There were several candidates, but the one that stood out the most was the model trained with the feature dataset on the SVM, which achieved a very good accuracy on the test data (96.25%) while having a very low inference time (0.007192 seconds). This made it a clear choice to implement this model in the final hand gesture recognition tool clear.

At last, the automatic hand gesture recognition tool was created. It consists of a two-component program designed to replace the computer's peripheral mouse control by a hand gesture control using the LM camera and the best model developed (SVM with 96.25% accuracy) to predict the hand gesture being performed.

7. SYSTEM'S VALIDATION PROTOCOL

This chapter describes the validation protocol of the automatic hand gesture recognition tool, aiming to assess the usability of the developed tool. This validation protocol followed an experimental study design and was carried out in two stages: first, a preliminary validation conducted in a laboratory environment with volunteers from the BiRDLab and secondly, a final validation conducted with surgeons (end-users) from the *Trofa Saúde Braga Centro* hospital. The participants and their characteristics are first specified. Then, the intervention is explained, followed by the data collection. Finally, the results are presented and discussed, and some conclusions on the usability of the application are made.

7.1 Participants

The study was approved by the institutional review board of the *Trofa Saúde*. All participants filled out an informed consent, based on the Helsinki declaration and the Oviedo convention, to participate in this study.

7.1.1 Preliminary validation

Eleven participants (gender: 7 males and 4 females, age: 25.73 ± 2.22, dominant hand: 0 left and 11 right, previous experience with touchless controls: 7 with no experience, 4 with experience in HTC Vive, 3 with experience in HoloLens, and 1 with experience in LM), recruited and admitted in the University of Minho, were enrolled in the study. There were no exclusion criteria. The Table 7.1 refers to the participants from BiRDLab that volunteered to perform this preliminary validation. The previous experience with touchless controls was asked to understand if it could be a factor on the results. As only one had experience with the LM, the sample size was too small to make any viable conclusions.

Table 7.1 - Age (years), gender (male/female), dominant hand (left/right) and previous experience with touchless controls of each participant on the preliminary validation

Participant	Age	Gender	Dominant Hand	Previous experience with touchless controls	
				Yes/No	Touchless Tools
Participant 1	24	Male	Right	No	-
Participant 2	26	Male	Right	No	-
Participant 3	23	Female	Right	No	-
Participant 4	22	Male	Right	No	-
Participant 5	27	Female	Right	No	-
Participant 6	27	Male	Right	No	-
Participant 7	24	Male	Right	No	-
Participant 8	27	Female	Right	Yes	HTC Vive, HoloLens
Participant 9	25	Female	Right	Yes	HTC Vice, HoloLens
Participant 10	29	Male	Right	Yes	HTC Vive
Participant 11	29	Male	Right	Yes	HTC Vive, HoloLens, LM

7.1.2 Final validation

Four participants, all surgeons (gender: 4 males and 0 females, age: 40.25 ± 6.37, dominant hand: 0 left and 4 right, previous experience with touchless controls: 4 with no experience), were enrolled in the study. There were no exclusion criteria. The Table 7.2 refers to the participants from the *Trofa Saúde Braga Centro* hospital that volunteered to perform the final validation.

Table 7.2 - Age (years), gender (male/female), dominant hand (left/right) and previous experience with touchless controls of each participant on the final validation

Participant	Age	Gender	Dominant Hand	Previous experience with touchless controls	
				Yes/No	Touchless Tools
Participant 1	39	Male	Right	No	-
Participant 2	36	Male	Right	No	-
Participant 3	35	Male	Right	No	-
Participant 4	51	Male	Right	No	-

7.2 Intervention protocol

This intervention was designed to assess the usability of the automatic hand gesture recognition tool. The intervention begins by explaining to the participants the six hand gestures available to control the computer and the corresponding action on the screen. The experimental study consisted of the following two phases: familiarization phase and data acquisition phase. In the familiarization phase, participants performed the validation protocol, first with the conventional mouse (1 trial) and then with the hand gesture recognition tool using the LM (1 trial). In the data acquisition phase, participants performed the same validation protocol, first with the conventional mouse (3 trials in a row), and then with the hand gesture recognition tool using the LM (3 trials in a row). Between each trial of the second phase, the preliminary validation participants had 1 minute to rest. The study was approved by the institutional review board of the *Trofa Saúde*, but due to the restrictions on the availability of the medical team, the time window allocated for the collection with each doctor led to a change in the protocol, namely a reduction from 3 trials to 1 trial in the data acquisition phase.

Figure 7.1 shows the validation protocol in detail, which consists of the following 8 actions: A) Click on the "Tools" button in the top left menu (cursor and click gestures); B) Click on the "Preoperative fiducial 3" button in the left menu (cursor and click gestures); C) Scroll through the slices in the top right corner of the screen until reaching the green dot referring to the preoperative fiducial 3 (zoom in and zoom out gestures); D) Rotate the bone in the middle of the screen until the green dot related to the preoperative fiducial 3 is seen (cursor and rotate gestures); E) Select the "Ruler" button in the bottom left menu (cursor and click gestures); F) Select the green point on the bone in the centre of the screen as accurately as possible (cursor and click gestures); G) Select the red point on the bone in the centre of the screen as accurately as possible (cursor and click gestures); H) Click the "Save file" button (cursor and click gestures).

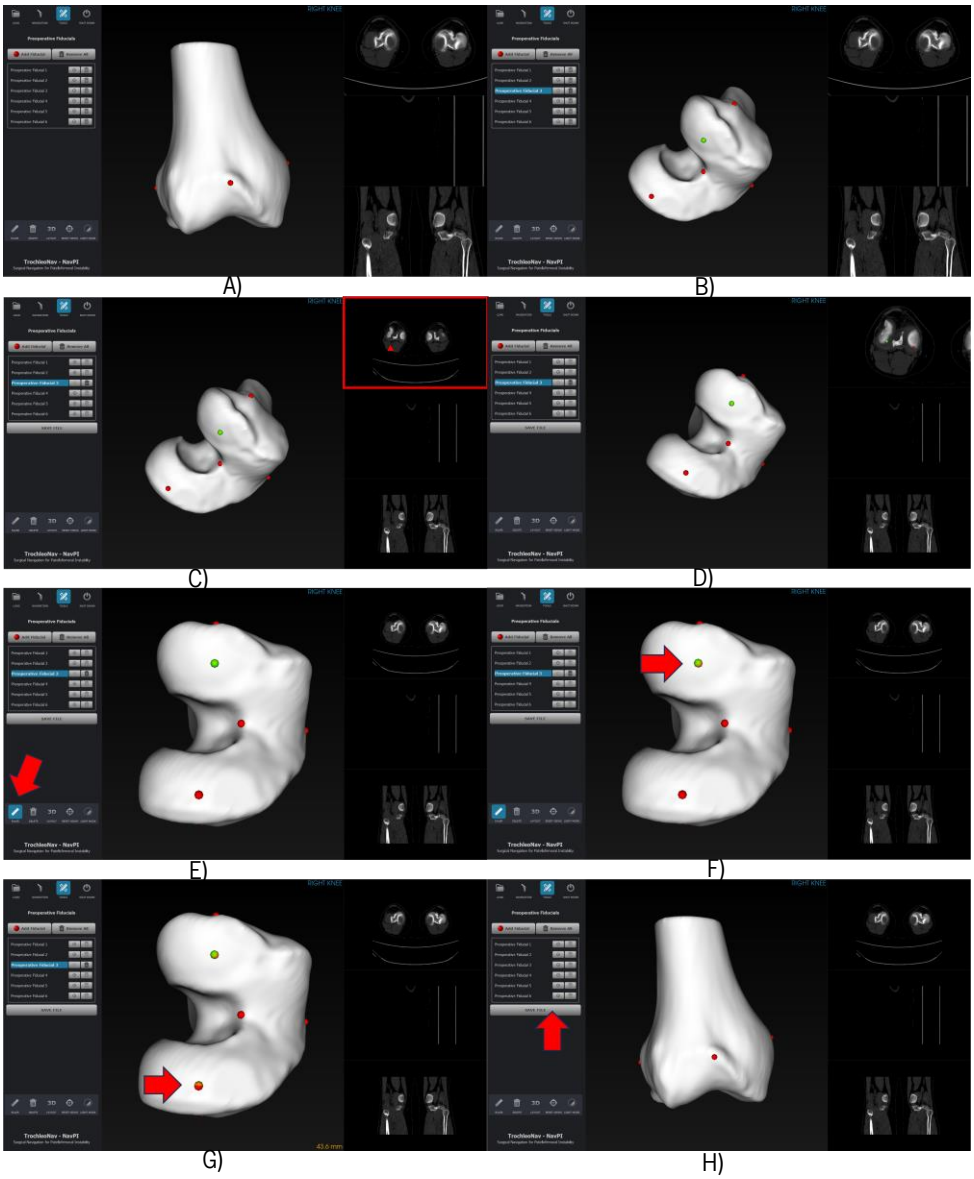


Figure 7.1 - The 8 defined actions for the intervention: (A) click tools button; (B) click preoperative fiducial 3 button; (C) scroll until find green dot; (D) rotate bone; (E) click ruler button; (F) select green dot; (G) select red dot; (H) click save file button.

7.3 Data collection

All participants completed the entire protocol. During the intervention, there were two quantitative evaluation metrics being collected: the time taken to complete the 8 defined actions, and the absolute error between the 3D position of the dots positioned by the user on the screen and the dots already in the bone. The aim of these results was to assess if the hand gesture recognition tool was able to achieve similar accuracy and time as a conventional mouse when performing the same tasks, given that the aim of this tool was to replace the mouse.

After the intervention, two evaluation metrics were used: one quantitative (SUS questionnaire) and one qualitative (user feedback). The SUS is a standardized test used to evaluate a wide variety of products and services, including software and applications, in terms of usability of the solution created [86]. It is

composed of 10 sentences that the participant must answer with one of the five available options ranging from 1 (Strongly Disagree) to 5 (Strongly Agree). This test produces a single score, which is the sum of the scores of the 10 sentences. The score contribution of each item will range from 0 to 4. For items 1,3,5,7 and 9, the score contribution is the scale position (1 to 5) minus 1. For items 2,4,6,8 and 10, the contribution is 5 minus the scale position (1 to 5). Finally, the sum of the scores is multiplied by 2.5 to obtain the overall system usability score [87]. A score above 68 is considered above average.

At last, the user's feedback is obtained. The participants were asked what needed the most improvement in the gesture recognition tool. This will allow to understand what should be tackled next and prepare the future work. In order to protect the privacy of the participants, the anonymity of their opinions and answers was guaranteed.

7.4 Results and discussion

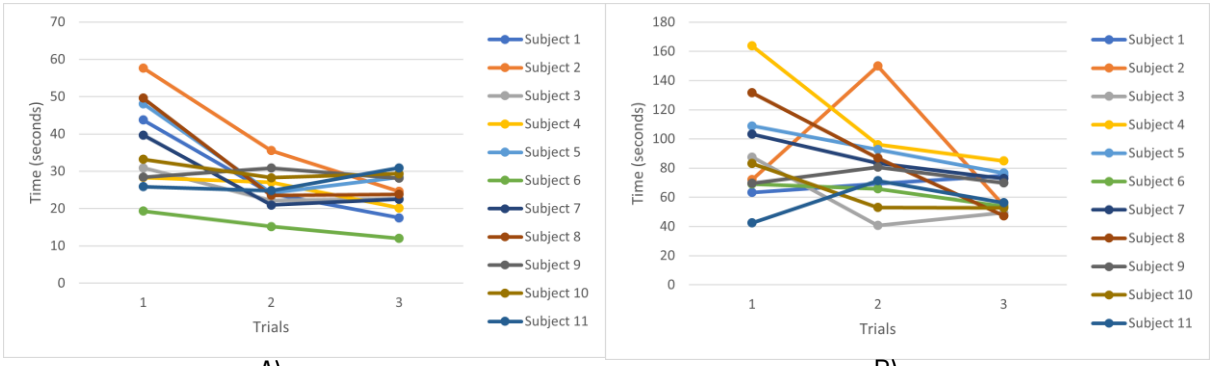
This subsection presents the results of the validation protocol for the preliminary and final groups. For each validation group, quantitative evaluation metrics, such as the time needed to complete the set of defined actions by both the mouse control and the hand gesture recognition tool, and the precision achieved by each form of control, and the results of the SUS test; and qualitative evaluation metrics, such as the users' feedback are presented.

7.4.1 Preliminary validation

Time taken

In an operating room environment, both the surgeon and the patient have an interest in minimizing the time taken to complete a surgical procedure. Therefore, the time required to perform a set of tasks in the software is very important. Thus, a good way to assess the operability and effectiveness of the developed hand gesture recognition tool is to benchmark it with the conventional control method, the mouse control. This allows a direct comparison between the two forms of control.

The average time taken to perform the planned set of actions using the traditional mouse control was 28.53 ± 9.76 seconds, and the average time taken for the same set of actions using the developed gesture recognition tool was 78.11 ± 28.02 seconds. This means that it takes 2.74 longer to complete the same task. Figure 7.2 presents the time taken by each volunteer for the three trials performed with the mouse and the hand gesture recognition tool, and some additional analysis can be made.



A) B)

Figure 7.2 - Time taken to complete the set of actions for the three trials: (A) with the mouse control; and (B) with the hand gesture recognition tool.

Analysing Figure 7.2, there is a lot of information to take in. Noticing that the scales of the two graphs are different, it can be seen that the participants took less time to complete the actions in the mouse control than in the control with the developed application. When looking more closely, the highest times in the mouse control are in the same range as the lowest times in the developed application control. These results were to be expected as people are very used to controlling the mouse and do so quite easily. However, although the participants took longer with the hand gesture recognition tool, they showed the same behaviour over time as with the mouse control, taking less time with each trial. This shows that as the participants worked with the tool, they became more comfortable with it and their results improved. This is consistent with the learning curve of any product/service, and it can be inferred that with more trials the time may continue to decrease until it converges to a time close to that of mouse control.

In addition to the learning curve, there is additional and future work that can be done to reduce the time even further, such as increasing the size of the interface buttons, adjusting the sensitivity of the mouse, and adjusting the sensitivity of the zoom in/out gestures.

Precision

While controlling surgical software, the handling precision can be very important. The desired task may be to place a screw in a specific location on a bone, rotate a bone to a specific view, or move a bone to a specific location on the screen, so it is important to evaluate whether the gesture recognition tool achieves the same precision as the mouse control. In a first approach, the precision of the click was chosen as an example to evaluate the precision of the tool. To achieve this, two tasks of the protocol were created with the sole interest of calculating the difference between the placed point and the existing point in the program, and then compare the results between the precision of the mouse and the gesture recognition tool. So, the smaller the difference, the greater the precision.

Table 7.3 presents the average distance (in millimetres) for the two points, for each participant on each form of control (mouse and gestures), in each trial. This was done to make the comparison easier and clearer to analyse.

Table 7.3 - Average distance (mm) for the two points, for each participant with the mouse control and with the hand gesture control, in each trial

Participants	**Form of control**	**Average distance (mm)**		
		Trial 1	**Trial 2**	**Trial 3**
Participant 1	Mouse	0.185 ± 0.077	0.089 ± 0.018	0.170 ± 0.131
	Hand gestures	0.963 ± 0.297	1.210 ± 0.418	0.611 ± 0.290
Participant 2	Mouse	0.223 ± 0.065	0.107 ± 0.041	0.159 ± 0.007
	Hand gestures	0.649 ± 0.160	1.27 ± 0.716	0.620 ± 0.260
Participant 3	Mouse	0.027 ± 0.020	0.126 ± 0.018	0.043 ± 0.030
	Hand gestures	0.400 ± 0.156	0.277 ± 0.142	0.399 ± 0.105
Participant 4	Mouse	0.069 ± 0.015	0.176 ± 0.032	0.126 ± 0.032
	Hand gestures	1.392 ± 0.499	2.396 ± 0.055	0.785 ± 0.388
Participant 5	Mouse	0.062 ± 0.033	0.242 ± 0.043	0.228 ± 0.104
	Hand gestures	0.993 ± 0.096	1.315 ± 0.525	0.280 ± 0.114
Participant 6	Mouse	0.055 ± 0.013	0.267 ± 0.090	0.181 ± 0.054
	Hand gestures	1.452 ± 0.490	2.038 ± 1.048	1.207 ± 0.200
Participant 7	Mouse	0.253 ± 0.157	0.353 ± 0.195	0.114 ± 0.004
	Hand gestures	1.366 ±0.048	1.664 ± 0.763	1.147 ± 0.138
Participant 8	Mouse	0.089 ± 0.028	0.071 ± 0.006	0.037 ± 0.011
	Hand gestures	1.136 ± 0.413	0.247 ± 0.005	0.985 ± 0.429
Participant 9	Mouse	0.145 ± 0.005	0.067 ± 0.006	0.074 ± 0.036
	Hand gestures	0.693 ± 0.110	0.589 ± 0.233	0.668 ± 0.020
Participant 10	Mouse	0.039 ± 0.013	0.116 ± 0.020	0.030 ± 0.011
	Hand gestures	0.974 ± 0.213	2.353 ± 0.854	0.384 ± 0.238
Participant 11	Mouse	0.157 ± 0.070	0.043 ± 0.010	0.101 ± 0.028
	Hand gestures	1.451 ± 0.126	1.325 ± 0.484	0.550 ± 0.241
Average	**Mouse**	**0.119 ± 0.074**	**0.151 ± 0.093**	**0.115 ± 0.062**
	Hand gestures	**1.043 ± 0.340**	**1.335 ± 0.716**	**0.694 ± 0.294**

As it can be seen on the averages shown in Table 7.3, the mouse control shows a better precision than the hand gesture recognition tool in all trials. Furthermore, the precision of the mouse control does not vary much from trial to trial, as it is already quite precise. These results were to be expected as people are very used to controlling the mouse and do so quite easily. However, with the hand gesture recognition tool, the participants showed more precision in the third trial than in the second and first trials, with some being much more precise. This shows that as the participants worked with the tool, they became more comfortable with it and their results improved. This is consistent with the learning curve of any product/service, and it can be inferred that with more trials the precision may continue to increase. In

addition to the learning curve, there is some work that can be done to help improve the precision, such as changing the sensitivity of the cursor when more precision is necessary.

SUS questionnaire

The SUS is a standardized test that will serve as an indicator of the user's perception of the quality and usability of the hand gesture recognition tool created. It will also serve as a baseline for future improvements of the tool and as a comparison point for similar tools developed by other studies. User feedback was also provided in response to the question "What needs the most improvement?". This question was created to find out if there is something that needs to be fixed urgently and to understand what the weaknesses of the developed solution were, from the point of view of the users, including the end-users. Table 7.4 shows the SUS questionnaire result for each participant in the preliminary validation group.

Table 7.4 - SUS questionnaire result for each participant in the preliminary validation group

Participants (P)	P 1	P 2	P 3	P 4	P 5	P 6	P 7	P 8	P 9	P 10	P 11
Score	70	80	85	75	52.5	77.5	82.5	87.5	80	70	77.5
Average score	**76.67 ± 9.86**										

The average of the results from the SUS questionnaire for the 11 participants in the preliminary validation group was 76.67 ± 9.86. Except for participant 5, the participants in this preliminary validation group have largely the same opinion about the usability of the hand gesture recognition tool. Tullis et al. [88] stated that the SUS test is the most reliable even with small sample sizes, which gives confidence in the obtained results. Furthermore, as this test exists for more than 25 years, there are more than 5 000 SUS observations, which allowed Sauro et al. [19] to create an approximate distribution of SUS scores in terms of percentile ranking. The paper established that the average score is 68 and that anything above this is considered above average and vice versa. Taking this into account, and considering the average score obtained by the participants, the usability of the developed tool can be considered above average. These results are positive and can be even better with further improvements. The response of each volunteer from the preliminary validation group to each of the 10 questions of the SUS questionnaire can be seen in Table 0.1 in Appendix I.

The SUS test provides an insight into the perceived usability of the tool but does not give the user the opportunity to express themselves freely about the tool, as the questions and answers in the questionnaire are fixed. Therefore, it seemed important to get a free and anonymous opinion from each

participant so that they did not feel pressured. To this end, participants were asked what they thought needed the most improvement. Most of the participants (9 out of 11) felt that the sensitivity of the cursor what needed the most work. This is understandable given that, when carrying out the protocol, it was possible to observe that most of the participants struggled and spent time placing the cursor in the desired location.

7.4.2 Final validation

Time taken

Table 7.5 presents the time taken by each surgeon with the mouse and with the hand gesture recognition tool. There is only one trial because, as mentioned above, time was a constraint, and the protocol was minimized as such.

Table 7.5 – Time taken by the final validation group to perform the tasks with the mouse control and with the hand gesture control

Participants	**Time on mouse control (seconds)**	**Time on gesture control (seconds)**
Participant 1	27.57	111.01
Participant 2	36.15	165.40
Participant 3	23.28	125.72
Participant 4	42.29	116.75
Average	**32.32 ± 7.39**	**129.72 ± 21.26**

The average time taken to perform the planned set of actions using the traditional mouse control was 32.32 seconds, and the average time taken for the same set of actions using the developed gesture recognition tool was 129.72 seconds. This means that it takes 4.01 times longer to complete the same task. As there is only one trial, there is no way to confirm that the times can decrease over time, but based on the data from the preliminary group, it is safe to assume that the same trend would be seen here, as the same learning curve would be expected. Despite the learning curve, the list of possible improvements given in the preliminary validation group still apply and could help reduce the times.

Precision

Table 7.6 presents the distance (in millimetres) between the two points placed by the user and the points on the screen, for each surgeon on each form of control (mouse and gestures).

Table 7.6 - Distance (mm) between the two points placed by the user and the points on the screen, for each surgeon with the mouse control and with the hand gesture control

Participants	Form of control	Distance to point 1 (mm)	Distance to point 2 (mm)	Average distance (mm)
Participant 1	Mouse	0.303	0.076	0.190 ± 0.114
	Hand gestures	0.934	0.905	0.920 ± 0.015
Participant 2	Mouse	0.081	0.195	0.138 ± 0.057
	Hand gestures	1.680	1.928	1.805 ± 0.124
Participant 3	Mouse	0.086	0.201	0.144 ± 0.058
	Hand gestures	1.599	0.597	1.099 ± 0.501
Participant 4	Mouse	0.096	0.053	0.075 ± 0.021
	Hand gestures	0.805	1.349	1.077 ± 0.272
Average	**Mouse**	**0.142 ± 0.094**	**0.132 ± 0.067**	
	Hand gestures	**1.255 ± 0.389**	**1.195 ± 0.501**	

Mouse control showed a better precision than the hand gesture control, just as in the preliminary group. Since there is only one trial, the surgeons did not have the same opportunity to get more accustomed to the hand gesture recognition tool like the preliminary group did. However, the learning curve observed in the preliminary group is expected to also apply in the final group, as the surgeons get more experience. Although the precision of the hand gesture recognition tool is likely to improve with experience, these results can be improved by adjusting the sensitivity of the cursor when necessary.

SUS questionnaire

Table 7.7 presents the SUS questionnaire result for each participant in the final validation group.

Table 7.7 - SUS questionnaire result for each participant in the final validation group

Participants	Participant 1	Participant 2	Participant 3	Participant 4
Score	75	60	62.5	72.5
Average score	**67.5 ± 6.37**			

The average of the results from the SUS questionnaire for the 4 surgeons in the final validation group was 67.5 ± 6.37. Rounding the mean value, it is at the limit of the average value determined by Sauro et al. [19], fulfilling the KPI defined in the objective 6 of this dissertation that stipulates that the results of this test should be above average. This value means that the surgeons found the developed hand gesture recognition tool less easy to use than the volunteers in the preliminary group. The sample size is small, which may not be a significant sample for drawing conclusions. However, as these are the end users of the tool, it gives a better impression of the quality of the work. Furthermore, these results

remain encouraging as it is already possible to identify some future work that could improve the overall experience of the gesture recognition tool. The response of each volunteer from the final validation group to each of the 10 questions of the SUS questionnaire can be seen in Table 0.2 in Appendix I.

To better understand the way forward, the surgeons were also asked about what needed the most improvement in the tool. One surgeon said "cursor speed", one said "cursor precision" and misclassification between gestures, one said "cursor sensitivity", and one said there was "nothing to improve". The first three answers are due to cursor sensitivity, meaning that most surgeons felt that the current cursor sensitivity was not right. As surgeons tend to have more precise hands due to the nature of their work, it was expected that the cursor control could be a problem, because they are used to achieving a high precision on a small working space. This follows the same feedback from the preliminary validation group and clearly shows that the cursor sensitivity is the issue that needs more attention. It is worth noting that all surgeons showed great acceptability and interest in the tool implemented and emphasised its potential interest in clinical practice.

7.5 Conclusions

The hand gesture recognition tool takes more time to complete the same set of defined tasks than the mouse control. However, there is a decreasing trend, which indicates that this time can be further reduced with more experience with the tool. The precision of the mouse control surpasses that of the hand gesture recognition tool. However, after only 3 trials (for participants in the preliminary validation group), the volunteers were more precise than in the first trial, indicating this precision tends to increase with greater familiarity with the developed tool.

In the preliminary validation, the results from the SUS questionnaire average at 76.67 ± 9.86. Except for one participant, the participants in the preliminary group have the same opinion about the usability of the developed tool. Since the average result for the SUS test is 68, the results obtained are above average and can be further improved. In the final validation, the results from the SUS questionnaire average at 67.5 ± 6.37. Rounding the mean value, this result is at the limit of the average results for the SUS test. The surgeons found the tool less easy to use than the preliminary group, but as these are the end users of the tool, they give a better impression of the quality of the work. These results are encouraging because it is possible to identify possible improvements thanks to the feedback from the users.

The feedback from the users on the preliminary validation and the final validation, defined the sensitivity of the cursor as the most needed improvement. This improvement combined with the learning curves mentioned earlier might improve the usability of the hand gesture recognition tool.

8. CONCLUSIONS

Globally, 310 million major surgeries are performed each year and during a medical procedure, there is a 2% to 5% chance of surgical site infections due to various reasons being one of them problems related to equipment sterilization. This means that between 6 200 000 and 15 500 000 people still suffer an unwanted infection that can negatively affect their physical and mental health. CAS is an emerging technology that typically relies on the control of a computer using a keyboard and mouse, where the surgeon takes advantage of technology for surgical planning and guidance that leads to better patient outcomes. Despite these advantages, the peripheral devices used to control the CAS systems (i.e., mouse and keyboard) contribute negatively to surgical site infections. The development of gesture recognition tools using AI models can help reduce the number of infections, reduce the time needed to sterilize the room for the next surgery and consequently increase the number of surgeries. Consequently, this dissertation designed, developed, and validated an automatic hand gesture recognition tool, based on a user-centred design, to help clinicians reduce the risk of contamination and in this way improve the quality of medical procedures.

The review on DL-based hand gesture recognition shows that there is some work currently being done but there is a lack of this application in the medical area. From the 29 studies included in this review the medical area is the third most studied with only 4 studies. The feature-based dataset was the most used with 26 studies, leaving only 3 articles that used the images given by the LM camera. There were a lot of architectures tested in these articles due to the high variability of the problems to solve and the several capabilities of each architecture. However, the most used architectures to train the models were the CNN with 11 studies and the NN with 7 studies. The most used metric to evaluate the performance of the model was accuracy, which was present in 26 studies. The studies reviewed showed a big lack of system validation. Only 6 studies presented some form of validation, but none offered a standardized test that could be further compared with other studies. Considering this, there is space to create a gesture recognition tool with a user-centred design that serves the end-user needs and has a standardized validation of the final system so that it can serve as a reference for future studies.

A protocol was performed with the end-users from the *Trofa Saúde Braga Centro* hospital that sought to understand what gestures were the most intuitive to perform a given set of actions. In this way, the developed tool has a design centred on the end user, as the hand gestures are decided by the end users. This fulfils the first clinical requirement and the third technological requirement defined in Chapter 3, advancing the literature review.

To have the necessary data to train AI models, a protocol was performed with 21 participants from the University of Minho, to create two balanced datasets (one with hand images and one with hand features) with the hand gestures defined in the previous protocol. This fulfils the first technological requirement defined in the literature review. It also creates a big and diverse dataset that enables the development of more robust and powerful models.

Then, these datasets were fed to several AI models in search of the one that could achieve the most performance. The best model developed using the SVM algorithm, with 96.25% accuracy on the test data and an inference time of 0.007192 seconds, was later implemented in the hand gesture recognition tool. This result gives the gesture recognition tool a good ability to do what it was designed for. This search for the best and most appropriate architecture for the defined gestures meets the second technological requirement.

The developed tool uses the LM camera to extract information from the hands, feeds it into the best model developed (SVM with 96.25% accuracy) and executes an action on the screen for the corresponding gesture predicted.

The validation protocol was carried out with two groups of validation: the preliminary validation performed on volunteers from the BiRDLab (11 participants), and the final validation performed on surgeons from the *Trofa Saúde Braga Centro* hospital (4 participants). The intervention consisted of completing a set of 8 actions on the mouse and the developed hand gesture recognition tool, aiming to assess the usability of the application developed as well as understand the differences between the mouse and the hand gesture control of the NavPI program. The results from the validation protocol showed that the hand gesture recognition tool took more time to complete a set of defined tasks than the mouse control, but showed a decreasing trend which indicates that this time can be further reduced with more experience with the tool. The precision of the mouse control surpasses the precision of the hand gesture recognition tool. However, with more experience, the volunteers were more precise than in the first trial, suggesting that they can become more precise as they become more familiar with the developed tool. The results of the SUS questionnaire carried out on the preliminary group showed that the tool was well accepted by the participants and that the perceived usability of the developed system is high. The average result of the final group is at the limit of the average for the SUS test, meaning that the surgeons found the developed less easy to use than the preliminary group. However, the insight from the surgeons is a valuable resource because they are the end users. This further shows the work done to build the hand gesture recognition tool around the end user and fulfils the fourth technological requirement, as well as the second clinical requirement.

The feedback from the two validation groups shows that the sensitivity of the cursor is the main complaint, and it should be improved in future works.

This dissertation allows to answer the RQs appointed in Chapter 1:

- **RQ1**: What specifications should be considered for the development of a LM hand gesture recognition application fitted for surgical navigation?

  Chapter 3 answered this RQ. The optimal LM hand gesture recognition application fitted for surgical navigation should consider technological and clinical requirements. The technological requirements are the following: 1) ensure a balanced dataset; 2) the model architecture should be defined based on the required gestures (static or dynamic); 3) define the number and which gestures are required; 4) define a validation protocol that includes real-life situations with the end-users. The clinical requirements are: 1) include end-user-centred gestures so that the end-user can manipulate digital medical images and 3D anatomical models as easily and intuitively as possible; 2) define a validation protocol to study the usability of the application.

- **RQ2**: What are the most appropriate gestures needed to control the LM application in a surgical environment?

  Chapter 5 answered this RQ. According to the results from the protocol performed with surgeons from the *Trofa Saúde Braga Centro* hospital, the most appropriate gestures to control the LM application in a surgical environment are gestures made with one hand, so as not to interfere with the potential use of a surgical tool with the other hand, and to allow manipulation of the surgical navigation software during surgery. Furthermore, the gestures are intuitive and some of them are based on actions performed daily on a personal phone (i.e., click, zoom in and zoom out). Ultimately, the set of selected gestures can be seen in Figure 5.1.

- **RQ3**: Can hand gesture recognition achieve good performance using AI algorithms for real-time use?

  Chapter 6 answered this RQ. During this thesis, several ML and DL models were developed, and their accuracy and inference time were evaluated. The 96.25% accuracy obtained from the SVM model on the test data, and the 0.007192 seconds of inference time showed that it is possible to achieve good performance on hand gesture recognition using AI algorithms for real time use. This model was trained on the feature-based dataset. In general, when trained with features, the models achieve better accuracies and lower inference times comparing to the models trained with the image dataset due to the differences in the architecture complexity.

The highest accuracy obtained from the models trained on the image-based dataset was 78.16% on the test data with an inference time of 0.24544 seconds.

- **RQ4**: Can a gesture recognition-based solution be suitable for controlling a surgical navigation application?

Chapter 7 answered this RQ. According to the time and precision metrics, the solution based on hand gesture recognition is worse compared to the traditional mouse control. However, these metrics improve over time and can be expected to continue to improve with more experience from the volunteers, following the learning curve of any product/service. According to the results of the SUS questionnaire, the usability of the solution is above average in the preliminary validation group and borderline in the final group. Both groups indicated that cursor sensitivity needed further optimization, so there is potential to improve the developed application. These results show that a gesture recognition-based solution can be suitable for controlling a surgical navigation application.

8.1 Future work

The future work comprises the following directions: (i) to change the two-dimensional approach of cursor control to a three-dimensional approach; (ii) to rethink the user interface of the surgical navigation application to be more prepared for the hand gesture control; (iii) to customize the cursor sensitivity according to the desired action on the screen; (iv) further develop AI models or improve the ones already studied in this dissertation; (v) use the confidence of a given prediction from a DL model as a threshold for the detection of a given hand gesture; (vi) to migrate the Python component to the C++ component, creating a more robust standalone solution for the problem; (vii) to study the effect on the quality of the developed tool on the newer version of the LM camera released in 2023; (viii) further validate the solution with surgeons to have very solid feedback with the end-users of the tool in order to make improvements to new versions of the tool as they emerge. Further work involves scientific dissemination of the achieved results in peer-review ISI/Scopus journals.

REFERENCES

[1] G. P. Dobson, "Trauma of major surgery: A global problem that is not going away," *International Journal of Surgery*, vol. 81, pp. 47–54, Sep. 2020, doi: 10.1016/J.IJSU.2020.07.017.

[2] A. Marie-Claude Roy and M. Stevens, "Guide to infection control in the healthcare setting the operating room," 2018, Accessed: Nov. 22, 2022. [Online]. Available: https://isid.org/wp-content/uploads/2019/06/ISID_GUIDE_THE_OPERATING_ROOM.pdf

[3] J. M. Badia, A. L. Casey, N. Petrosillo, P. M. Hudson, S. A. Mitchell, and C. Crosby, "Impact of surgical site infection on healthcare costs and patient outcomes: a systematic review in six European countries," *Journal of Hospital Infection*, vol. 96, no. 1, pp. 1–15, May 2017, doi: 10.1016/j.jhin.2017.03.004.

[4] M. Richter *et al.*, "Computer-assisted surgery (CAS) based correction of posttraumatic ankle and hindfoot deformities—Preliminary results," *Foot and Ankle Surgery*, vol. 12, no. 3, pp. 113–119, Jan. 2006, doi: 10.1016/J.FAS.2006.02.003.

[5] "Medical Robotics Market & Computer-assisted Surgery Report." Accessed: Dec. 26, 2022. [Online]. Available: https://www.bccresearch.com/market-research/healthcare/medical-robotics-mrcas-market.html

[6] S. Jaemsuwan, S. Arunjaroensuk, B. Kaboosaya, K. Subbalekha, N. Mattheos, and A. Pimkhaokham, "Comparison of the accuracy of implant position among freehand implant placement, static and dynamic computer-assisted implant surgery in fully edentulous patients: a non-randomized prospective study," *Int J Oral Maxillofac Surg*, Jun. 2022, doi: 10.1016/J.IJOM.2022.05.009.

[7] D. Apostolakis, G. Michelinakis, P. Kamposiora, and G. Papavasiliou, "The current state of computer assisted orthognathic surgery: A narrative review," *J Dent*, vol. 119, p. 104052, Apr. 2022, doi: 10.1016/J.JDENT.2022.104052.

[8] A. Mandelberger, K. Neal, R. Bueser, and M. Nimaroff, "2529 Increased Surgical Site Infections in Robotic Hysterectomies in a Large Health System," *J Minim Invasive Gynecol*, vol. 26, no. 7, p. S216, Nov. 2019, doi: 10.1016/j.jmig.2019.09.454.

[9] A. F. Botero-Ospina, S. I. Duque-Vallejo, J. F. Ochoa-Gómez, and A. M. Hernández-Valdivieso, "Touchless control module for diagnostic images at the surgery room using the Leap Motion system and 3D Slicer software," *Revista Facultad de Ingeniería Universidad de Antioquia*, vol. 2017, no. 82, pp. 40–46, Mar. 2017, doi: 10.17533/UDEA.REDIN.N82A05.

[10] J. Pauchot *et al.*, "Leap Motion Gesture Control With Carestream Software in the Operating Room to Control Imaging: Installation Guide and Discussion," *Surg Innov*, vol. 22, no. 6, pp. 615–620, Dec. 2015, doi: 10.1177/1553350615587992.

[11] S. Mitra, S. Member, and T. Acharya, "Gesture Recognition: A Survey," *APPLICATIONS AND REVIEWS*, vol. 37, no. 3, 2007, doi: 10.1109/TSMCC.2007.893280.

[12] M. A. Anusuya and S. K. Katti, "Speech Recognition by Machine: A Review," *IJCSIS) International Journal of Computer Science and Information Security*, vol. 6, no. 3, 2009, doi: 10.48550/arxiv.1001.2267.

[13] S. Mauser and O. Burgert, "Touch-Free, Gesture-Based Control of Medical Devices and Software Based on the Leap Motion Controller," *Stud Health Technol Inform*, vol. 196, pp. 265–270, 2014, doi: 10.3233/978-1-61499-375-9-265.

[14] J. P. Wachs *et al.*, "A Gesture-based Tool for Sterile Browsing of Radiology Images," *J Am Med Inform Assoc*, vol. 15, no. 3, p. 321, Feb. 2008, doi: 10.1197/JAMIA.M241.

[15] S. Ameur, A. Ben Khalifa, and M. S. Bouhlel, "Hand-Gesture-Based Touchless Exploration of Medical Images with Leap Motion Controller," in *2020 17th International Multi-Conference on Systems, Signals & Devices (SSD)*, IEEE, 2020. Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9364244/

[16] A. reum Lee, Y. Cho, S. Jin, and N. Kim, "Enhancement of surgical hand gesture recognition using a capsule network for a contactless interface in the operating room," *Comput Methods Programs Biomed*, vol. 190, Jul. 2020, doi: 10.1016/j.cmpb.2020.105385.

[17] X. Li, R. Wang, B. Zhang, and K. Wang, "Design of VR Experimental System Based on Leap Motion Gesture Recognition," in *2022 3rd International Conference on Electronic Communication and Artificial Intelligence (IWECAI)*, IEEE, 2022. Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9750795/

[18] S. Tripathy, R. Sahoo, A. K. Dash, and D. P. Dogra, "Natural Gestures to Interact with 3D Virtual Objects using Deep Learning Framework," in *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*, IEEE, 2019. Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/8929637/

[19] J. Sauro, "A Practical Guide to the System Usability Scale: Background, Benchmarks & Best Practices," p. 162, 2011, Accessed: Oct. 03, 2023. [Online]. Available: http://books.google.com/books/about/A_Practical_Guide_to_the_System_Usabilit.html?id=BL0kKQEACAAJ&pgis=1

[20] A. M. Turing, “COMPUTING MACHINERY AND INTELLIGENCE,” *Computing Machinery and Intelligence. Mind*, vol. 49, pp. 433–460, 1950.

[21] “What is Machine Learning? | IBM.” Accessed: Jul. 10, 2023. [Online]. Available: https://www.ibm.com/topics/machine-learning

[22] E. Alpaydin, *Introduction to Machine Learning*, Fourth. MIT, 2020.

[23] “Semi-Supervised Learning... the great unknown.” Accessed: Oct. 29, 2023. [Online]. Available: https://telefonicatech.com/en/blog/semi-supervised-learning-the-great-unknown

[24] “What are Neural Networks? | IBM.” Accessed: Jul. 27, 2023. [Online]. Available: https://www.ibm.com/topics/neural-networks

[25] “Artificial neural network - Wikipedia.” Accessed: Oct. 29, 2023. [Online]. Available: https://en.wikipedia.org/wiki/Artificial_neural_network

[26] “About Linear Regression | IBM.” Accessed: Jul. 27, 2023. [Online]. Available: https://www.ibm.com/topics/linear-regression

[27] “What is Logistic regression? | IBM.” Accessed: Jul. 27, 2023. [Online]. Available: https://www.ibm.com/topics/logistic-regression

[28] “Linear regression.” Accessed: Oct. 29, 2023. [Online]. Available: https://www.jeremyjordan.me/linear-regression/

[29] “Logistic Regression Explained. [—Logistic Regression explained... | by James Thorn | Towards Data Science.” Accessed: Oct. 29, 2023. [Online]. Available: https://towardsdatascience.com/logistic-regression-explained-9ee73cede081

[30] “Spotfire | Demystifying the Random Forest Algorithm for Accurate Predictions.” Accessed: Oct. 29, 2023. [Online]. Available: https://www.spotfire.com/glossary/what-is-a-random-forest

[31] I. T. Jollife and J. Cadima, “Principal component analysis: a review and recent developments,” *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, Apr. 2016, doi: 10.1098/RSTA.2015.0202.

[32] “Biplot of principal component analysis | U.S. Geological Survey.” Accessed: Oct. 29, 2023. [Online]. Available: https://www.usgs.gov/media/images/biplot-principal-component-analysis

[33] S. Shukla, “A Review ON K-means DATA Clustering APPROACH,” *International Journal of Information & Computation Technology*, vol. 4, pp. 1847–1860, 2014, Accessed: Jul. 27, 2023. [Online]. Available: http://www.irphouse.com

[34] “K means Clustering - Introduction - GeeksforGeeks.” Accessed: Oct. 29, 2023. [Online]. Available: https://www.geeksforgeeks.org/k-means-clustering-introduction/

[35] T. J. M., “Models for machine learning - IBM Developer.” Accessed: Sep. 17, 2023. [Online]. Available: https://developer.ibm.com/articles/cc-models-machine-learning/#reinforcement-learning

[36] C. Bisogni, L. Cimmino, M. De Marsico, F. Hao, and F. Narducci, “Emotion recognition at a distance: The robustness of machine learning based on hand-crafted facial features vs deep learning models,” *Image Vis Comput*, vol. 136, p. 104724, Aug. 2023, doi: 10.1016/J.IMAVIS.2023.104724.

[37] S. Madhavan and M. T. Jones, “Deep learning architectures - IBM Developer.” Accessed: Jul. 08, 2023. [Online]. Available: https://developer.ibm.com/articles/cc-machine-learning-deep-learning-architectures/

[38] L. Alzubaidi *et al.*, “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions,” *Journal of Big Data 2021 8:1*, vol. 8, no. 1, pp. 1–74, Mar. 2021, doi: 10.1186/S40537-021-00444-8.

[39] “CNN for Deep Learning | Convolutional Neural Networks.” Accessed: Oct. 29, 2023. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/

[40] B. Lindemann, T. Müller, H. Vietz, N. Jazdi, and M. Weyrich, “A survey on long short-term memory networks for time series prediction,” *Procedia CIRP*, vol. 99, pp. 650–655, Jan. 2021, doi: 10.1016/J.PROCIR.2021.03.088.

[41] K. Cho *et al.*, “Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation”.

[42] “A Brief Introduction to Recurrent Neural Networks | by Jonte Dancker | Towards Data Science.” Accessed: Oct. 29, 2023. [Online]. Available: https://towardsdatascience.com/a-brief-introduction-to-recurrent-neural-networks-638f64a61ff4

[43] Y. Chen, M. Mancini, X. Zhu, and Z. Akata, “Semi-Supervised and Unsupervised Deep Visual Learning: A Survey,” *IEEE Trans Pattern Anal Mach Intell*, pp. 1–23, Aug. 2022, doi: 10.1109/tpami.2022.3201576.

[44] S. Lakshminarayanan, “Application of Self-Organizing Maps on Time Series Data for identifying interpretable Driving Manoeuvres,” *European Transport Research Review*, vol. 12, no. 1, pp. 1–11, Dec. 2020, doi: 10.1186/S12544-020-00421-X/FIGURES/16.

[45] “Self-organizing maps | viscovery.net.” Accessed: Oct. 29, 2023. [Online]. Available: https://www.viscovery.net/self-organizing-maps

[46] W. H. Lopez Pinaya, S. Vieira, R. Garcia-Dias, and A. Mechelli, "Autoencoders," *Machine Learning: Methods and Applications to Brain Disorders*, pp. 193–208, Mar. 2020, doi: 10.1016/B978-0-12-815739-8.00011-0.

[47] "2019.12.09(pm): Autoencoder - SEONGJUHONG." Accessed: Oct. 29, 2023. [Online]. Available: https://seongjuhong.com/2019-12-09pm-autoencoder/

[48] Y. SHI, Y. LI, X. FU, M. I. A. O. Kaibin, and M. I. A. O. Qiguang, "Review of dynamic gesture recognition," *Virtual Reality & Intelligent Hardware*, vol. 3, no. 3, pp. 183–206, Jun. 2021, doi: 10.1016/J.VRIH.2021.05.001.

[49] D. Hirafuji Neiva and C. Zanchettin, "Gesture recognition: A review focusing on sign language in a mobile context," *Expert Syst Appl*, vol. 103, pp. 159–183, Aug. 2018, doi: 10.1016/J.ESWA.2018.01.051.

[50] S. B. Abdullahi and K. Chamnongthai, "American Sign Language Words Recognition of Skeletal Videos Using Processed Video Driven Multi-Stacked Deep LSTM," *Sensors*, vol. 22, no. 4, Feb. 2022, doi: 10.3390/s22041406.

[51] S. B. Abdullahi and K. Chamnongthai, "American Sign Language Words Recognition Using Spatio-Temporal Prosodic and Angle Features: A Sequential Learning Approach," *IEEE Access*, vol. 10, 2022, Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9702061/

[52] S. Ameur, A. Ben Khalifa, and M. S. Bouhlel, "A novel hybrid bidirectional unidirectional LSTM network for dynamic hand gesture recognition with Leap Motion," *Entertain Comput*, vol. 35, Aug. 2020, doi: 10.1016/J.ENTCOM.2020.100373.

[53] H. Brock, J. P. Chulani, L. Merino, D. Szapiro, and R. Gomez, "Developing a Lightweight Rock-Paper-Scissors Framework for Human-Robot Collaborative Gaming," *IEEE Access*, vol. 8, 2020, Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9239276/

[54] A. Caputo *et al.*, "SFINGE 3D: A novel benchmark for online detection and recognition of heterogeneous hand gestures from 3D fingers' trajectories," *Computers and Graphics (Pergamon)*, vol. 91, pp. 232–242, Oct. 2020, doi: 10.1016/J.CAG.2020.07.014.

[55] D. Enikeev and S. Mustafina, "Russian Fingerspelling Recognition Using Leap Motion Controller," in *2021 3rd International Conference on Control Systems, Mathematical Modeling, Automation and Energy Efficiency (SUMMA)*, IEEE, 2021. Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9632152/

[56] B. Hu and J. Wang, “Deep Learning Based Hand Gesture Recognition and UAV Flight Controls,” in *2018 24th International Conference on Automation and Computing (ICAC)*, IEEE, 2018. Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/8748953/

[57] B. Hu and J. Wang, “Deep Learning Based Hand Gesture Recognition and UAV Flight Controls,” *INTERNATIONAL JOURNAL OF AUTOMATION AND COMPUTING*, vol. 17, no. 1, SI, pp. 17–29, Feb. 2020, doi: 10.1007/s11633-019-1194-7.

[58] A. Ikram and Y. Liu, “Real Time Hand Gesture Recognition Using Leap Motion Controller Based on CNN-SVM Architechture,” in *2021 IEEE 7th International Conference on Virtual Reality (ICVR)*, IEEE, 2021. Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9483844/

[59] Z. Katılmış and C. Karakuzu, “ELM based two-handed dynamic Turkish Sign Language (TSL) word recognition,” *Expert Syst Appl*, vol. 182, Nov. 2021, doi: 10.1016/J.ESWA.2021.115213.

[60] K. Kritsis, M. Kaliakatsos-Papakostas, V. Katsouros, and A. Pikrakis, “Deep Convolutional and LSTM Neural Network Architectures on Leap Motion Hand Tracking Data Sequences,” in *2019 27th European Signal Processing Conference (EUSIPCO)*, IEEE, 2019. Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/8902973/

[61] C. K. M. Lee, K. K. H. Ng, C. H. Chen, H. C. W. Lau, S. Y. Chung, and T. Tsoi, “American sign language recognition and training method with recurrent neural network,” *Expert Syst Appl*, vol. 167, Apr. 2021, doi: 10.1016/J.ESWA.2020.114403.

[62] J. Li, J. Zhong, F. Chen, and C. Yang, “An Incremental Learning Framework for Skeletal-based Hand Gesture Recognition with Leap Motion,” in *2019 IEEE 9th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, IEEE, 2019. Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9066761/

[63] Y.-W. Lin, L.-H. Kao, and C.-T. Yeh, “FingerTalk: Real-time Fingertip Trajectories Tracking Service based on IoTtalk,” in *2020 21st Asia-Pacific Network Operations and Management Symposium (APNOMS)*, IEEE, 2020. Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9237035/

[64] H. Liu, T. Fang, T. Zhou, Y. Wang, and L. Wang, “Deep Learning-based Multimodal Control Interface for Human-Robot Collaboration,” *Procedia CIRP*, vol. 72, pp. 3–8, 2018, doi: 10.1016/J.PROCIR.2018.03.224.

[65] A. Mittal, P. Kumar, P. P. Roy, R. Balasubramanian, and B. B. Chaudhuri, “A Modified LSTM Model for Continuous Sign Language Recognition Using Leap Motion,” *IEEE Sens J*, vol. 19, no.

16, 2019, Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/8684245/

[66] A. B. Oktay and A. Kocer, "Differential diagnosis of Parkinson and essential tremor with convolutional LSTM networks," *Biomed Signal Process Control*, vol. 56, Feb. 2020, doi: 10.1016/j.bspc.2019.101683.

[67] A. A. Tyutyunnik, E. I. Lobaneva, and A. I. Lazarev, "Contactless access control system for critical objects based on deep learning neural networks," in *2021 3rd International Youth Conference on Radio Electronics, Electrical and Power Engineering (REEPE)*, IEEE, 2021. Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9388077/

[68] Z. Wang and J. Zhang, "Continuous Sign Language Recognition based on Multi-Part Skeleton Data," in *2021 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2021. Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9534003/

[69] B.-X. Wu, C.-G. Yang, and J.-P. Zhong, "Research on Transfer Learning of Vision-based Gesture Recognition," *INTERNATIONAL JOURNAL OF AUTOMATION AND COMPUTING*, vol. 18, no. 3, pp. 422–431, Jun. 2021, doi: 10.1007/s11633-020-1273-9.

[70] S. Yamamoto, S. Ito, M. Ito, and M. Fukumi, "Authentication of Aerial Input Numerals by Leap Motion and CNN," in *2018 IEEE International Conference on Internet of Things and Intelligence System (IOTAIS)*, IEEE, 2018. Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/8600847/

[71] S. Yamamoto, M. Fukumi, S.-I. Ito, and M. Ito, "Recognition of Aerial Input Numerals by Leap Motion and CNN," in *2018 Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS)*, IEEE, 2018. Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/8716241/

[72] J. Yang and R. Horie, "An improved computer interface comprising a recurrent neural network and a natural user interface," *Procedia Comput Sci*, vol. 60, no. 1, pp. 1386–1395, 2015, doi: 10.1016/J.PROCS.2015.08.213.

[73] F. Yasir, P. W. C. Prasad, A. Alsadoon, A. Elchouemi, and S. Sreedharan, "Bangla Sign Language recognition using convolutional neural network," in *2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, IEEE, 2017. Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/8342533/

[74] M. Zhang, Y. Zheng, C. Cui, Z. Meng, and C.-M. Own, "Preliminary Application of Gesture Recognition to Virtual Acupuncture," in *2017 10th International Symposium on Computational Intelligence and Design (ISCID)*, IEEE, 2017. Accessed: Oct. 04, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/8275791/

[75] A. P. Wibawa, A. B. P. Utama, H. Elmunsyah, U. Pujianto, F. A. Dwiyanto, and L. Hernandez, "Time-series analysis with smoothed Convolutional Neural Network," *J Big Data*, vol. 9, no. 1, pp. 1–18, Dec. 2022, doi: 10.1186/S40537-022-00599-Y/TABLES/12.

[76] M. Hüsken and P. Stagge, "Recurrent neural networks for time series classification," *Neurocomputing*, vol. 50, pp. 223–235, 2003, doi: 10.1016/S0925-2312(01)00706-8.

[77] N. Japkowicz, "Why Question Machine Learning Evaluation Methods? (An illustrative review of the shortcomings of current methods)," 2006.

[78] C. Tepe and M. C. Demir, "Real-Time Classification of EMG Myo Armband Data Using Support Vector Machine," *IRBM*, vol. 43, no. 4, pp. 300–308, Aug. 2022, doi: 10.1016/J.IRBM.2022.06.001.

[79] "Leap Motion Controller TM", Accessed: Jul. 12, 2023. [Online]. Available: https://www.ultraleap.com/datasheets/Leap_Motion_Controller_Datasheet.pdf

[80] F. Weichert, D. Bachmann, B. Rudak, and D. Fisseler, "Analysis of the Accuracy and Robustness of the Leap Motion Controller," *Sensors 2013, Vol. 13, Pages 6380-6393*, vol. 13, no. 5, pp. 6380–6393, May 2013, doi: 10.3390/S130506380.

[81] "How Hand Tracking Works | Ultraleap." Accessed: Jul. 12, 2023. [Online]. Available: https://www.ultraleap.com/company/news/blog/how-hand-tracking-works/

[82] "Leap Concepts - Ultraleap documentation." Accessed: Jul. 12, 2023. [Online]. Available: https://docs.ultraleap.com/tracking-api/leapc-guide/leap-concepts.html

[83] L. Shao, "Hand Tracking With Leap Motion Controller", Accessed: Jul. 12, 2023. [Online]. Available: https://www.leapmotion.com/

[84] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, Sep. 2014, Accessed: Sep. 19, 2023. [Online]. Available: https://arxiv.org/abs/1409.1556v6

[85] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition", Accessed: Sep. 19, 2023. [Online]. Available: http://image-net.org/challenges/LSVRC/2015/

[86] “System Usability Scale (SUS) | Usability.gov.” Accessed: Oct. 03, 2023. [Online]. Available: https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html

[87] J. Brooke, “SUS: A quick and dirty usability scale,” *Usability Eval. Ind.*, vol. 189, Nov. 1995.

[88] T. Tullis and J. Stetson, “A Comparison of Questionnaires for Assessing Website Usability,” Jun. 2006.

APPENDIX I

Table 0.1 and Table 0.2 show the answer of each participant in the preliminary group and final group, respectively, to each of the 10 questions from the SUS questionnaire. The possible answers were: TD - totally disagree, D - disagree, N - neutral, A – agree, and TA - totally agree.

Table 0.1 – Response of each volunteer in the preliminary group to each of the 10 questions from the SUS questionnaire

	Participants										
Questions	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
1. I think that I would like to use this system frequently.	A	A	A	N	A	N	TA	A	A	A	TA
2. I found the system unnecessarily complex.	N	D	TD	TD	N	D	TD	D	TD	D	D
3. I thought the system was easy to use.	N	A	A	A	D	D	A	A	A	A	A
4. I think that I would need the support of a technical person to be able to use this system.	D	A	D	D	A	D	N	TD	TD	A	D
5. I found the various functions in this system were well integrated.	A	TA	TA	TA	A	TA	TA	TA	A	A	A
6. I thought there was too much inconsistency in this system.	D	TD	TD	TD	D	TD	TD	TD	D	D	D
7. I would imagine that most people would learn to use this system very quickly.	N	TA	TA	N	N	A	TA	TA	A	A	TA
8. I found the system very cumbersome to use.	D	TD	D	TD	D	TD	D	D	D	D	D
9. I felt very confident using the system.	A	A	A	N	N	A	N	TA	A	A	A
10. I needed to learn a lot of things before I could get going with this system.	TD	D	D	N	A	TD	D	D	D	D	TD

Table 0.2 - Response of each volunteer in the final group to each of the 10 questions from the SUS questionnaire

	Participants			
Questions	**1**	**2**	**3**	**4**
1. I think that I would like to use this system frequently.	A	A	A	A
2. I found the system unnecessarily complex.	D	N	D	D
3. I thought the system was easy to use.	A	A	A	N
4. I think that I would need the support of a technical person to be able to use this system.	D	A	A	D
5. I found the various functions in this system were well integrated.	A	A	A	A
6. I thought there was too much inconsistency in this system.	D	D	D	D
7. I would imagine that most people would learn to use this system very quickly.	A	N	D	A
8. I found the system very cumbersome to use.	D	N	D	D
9. I felt very confident using the system.	A	N	N	N
10. I needed to learn a lot of things before I could get going with this system.	D	D	D	TD