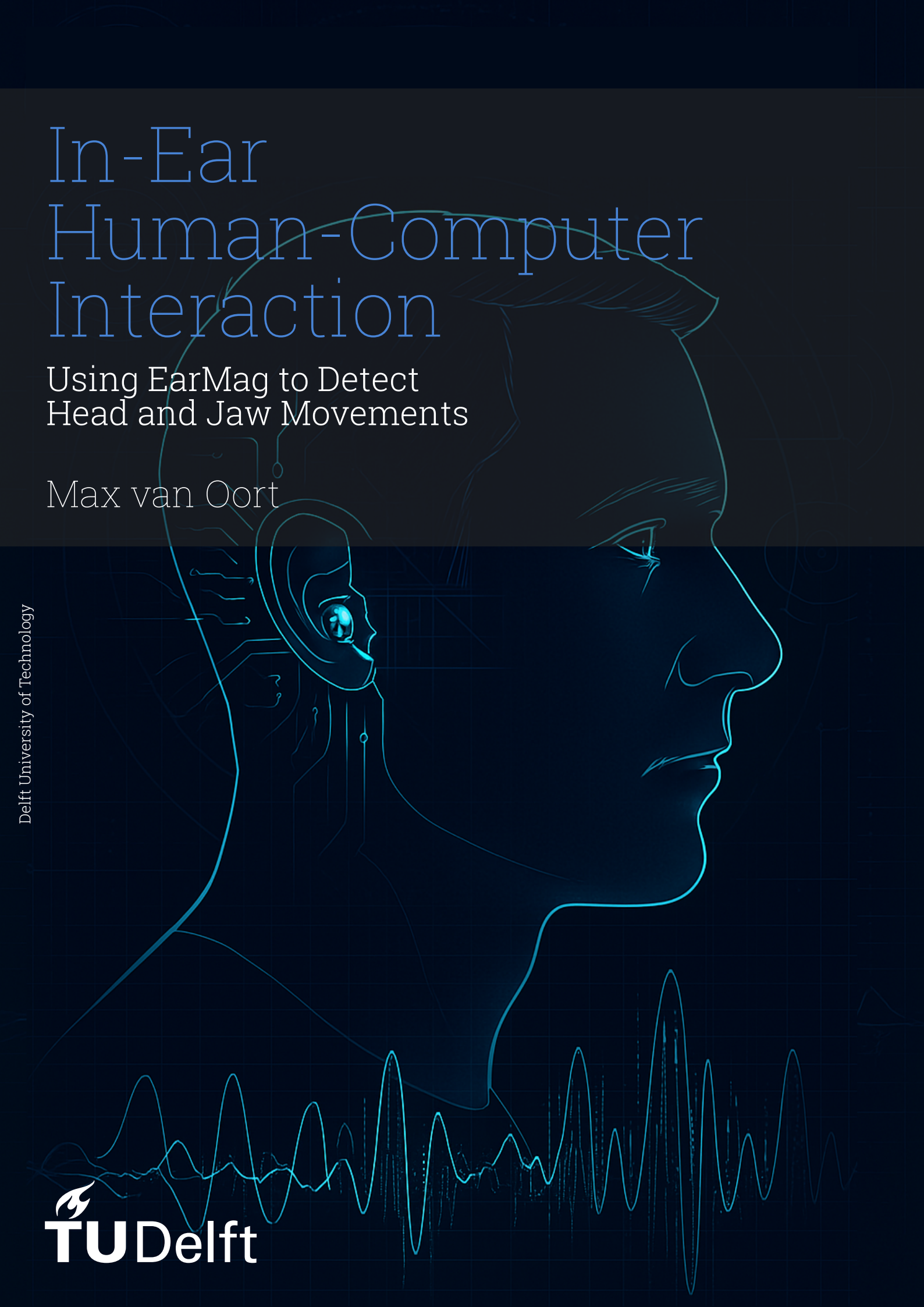
In-Ear Human-Computer Interaction

Using EarMag to Detect Head and Jaw Movements

Max van Oort



Delft University of Technology

TU Delft

In-Ear Human-Computer Interaction

Using EarMag to Detect Head and Jaw Movements

by

Max van Oort

to obtain the degree of Master of Science

at the Delft University of Technology,

to be defended publicly on Friday August 29, 2025 at 09:30 AM.

Student number:	5113164	
Project duration:	April 22, 2024 – June 30, 2025	
Thesis committee:	Przemysław Pawełczak,	TU Delft, thesis supervisor
	Ujwal Gadiraju,	TU Delft, committee member
	Gabriel Sáenz,	SOVN, supervisor at the company

A shortened version of this thesis was published as a workshop paper:

Max J. F. van Oort, Gabriel E. Sáenz, Selina Tirtajana, and Przemysław Pawełczak. 2025. *EarMag: In-Ear Magnetosensing for Jaw and Head Gesture-Based Human-Computer Interaction.* In *Proceedings of the 6th International Workshop on Earable Computing (EarComp '25), co-located with the 2025 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp Companion '25)*, October 12–16, 2025, Espoo, Finland. https://doi.org/10.1145/3714394.3757254

Cover:	In-Ear Human-Computer Interaction, created with ChatGPT.
Note:	ChatGPT was also used to assist in coding parts of the data processing pipeline.

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Preface

This thesis was written to conclude the Master Computer and Embedded Systems Engineering at the Delft University of Technology. For the last year, I have been following an internship at SOVN, who offered me their earables to investigate the feasibility of in-ear human-computer interaction.

While writing this thesis, I assumed that the reader would have a basic understanding of the traditional machine learning framework and the concept of in-ear human-computer interaction.

Readers who are not familiar with current in-ear human activity recognition systems can find these in Chapter 2. Readers with a special interest in the data collection procedure can find the study design in Chapter 3. Readers who are particularly interested in the methods or the results can find them in Chapters 4, 5 respectively.

I would like to thank my supervisor at the university Przemysław Pawełczak for his feedback and guidance throughout the project. I am also very thankful for the feedback and support from my daily supervisor at the company Gabriel Sáenz. I would also like to thank Vivian Dsouza for his early contributions in this project. Furthermore, I am very grateful for the feedback and inspiration of orofacial physiotherapist Simone Gouw. I also want to thank all the participants who voluntarily helped me with the data collection. In return for helping me with the data collection, I gave the volunteers a freshly baked red velvet cupcake, since I like to bake when I was not working on the project. Lastly, I would like to thank the rest of the SOVN team for their feedback during the weekly stand-up meetings and for participating in the team activities (dinner, bowling, billiards), where I really felt part of the team.

Max van Oort
Delft, August 2025

Summary

Human-Computer Interaction (HCI) has long been an active area of research, with continual advancements in sensing techniques enabling new, hands-free ways for humans to interact with computers. Among these, earable sensing is gaining attention as a versatile and unobtrusive approach.

This thesis investigates In-Ear Magnetosensing (EarMag) as a novel sensing technique for detecting jaw and head movements in the context of HCI. While previous work has explored acoustic sensing, inertial measurement units, and facial video, the potential of EarMag as a fine-grained, non-invasive sensing technique remains to be explored.

As a proof of concept, data from 17 orofacial physiotherapy-related exercises was collected using EarMag-enabled earables worn by 21 participants. Various signal processing techniques were applied to reduce noise and drift in the magnetic data. From this filtered data, ten key features were extracted and used to train and evaluate multiple machine learning models.

A soft voting ensemble classifier, combining a Support Vector Machine and a Random Forest, was trained to recognize five selected exercises. Validation on a test set of ten users, excluded from training, yielded a classification accuracy of 76%, demonstrating the feasibility of using EarMag for real-time movement detection in HCI.

This research highlights the potential of EarMag for hands-free interaction applications, including assistive technologies, silent speech interfaces, and biosignal tracking. However, challenges remain: individual anatomical differences in ear shape significantly impact the sensor readings, leading to variation in model performance between users. Additionally, the models in this study were limited to classical machine learning techniques, while future work may benefit from exploring deep learning approaches for improved generalization.

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
HCI	Human-Computer Interaction
EarMag	In-Ear Magnetosensing
IMU	Inertial Measurement Unit
TMJ	Temporomandibular Joint
GUI	Graphical User Interface
TSFEL	Time Series Feature Extraction Library
RFE	Recursive Feature Elimination
SNR	Signal-to-Noise Ratio
SOVM	Sum Of Vector Magnitudes
SVM	Support Vector Machine
LR	Logistic Regression
RF	Random Forest
XGBoost	eXtreme Gradient Boosting
SD	Standard Deviation
CV	Coefficient of Variation
Var	Variability
LOSO	Leave-One-Subject-Out
CWT	Continuous Wavelet Transform
UUID	Universal Unique Identifier

1

Introduction

In recent years, the field of Human-Computer Interaction (HCI) has experienced significant advances in computing power, computer vision, and machine learning techniques [1]. Hands- and eye-free HCI can be established by placing sensors on different parts of the body including the ear, while the anatomical properties and location of the ear create the most unique sensing opportunities. Earables, earbuds with a set of sensors, are widely used to fill in the sensing opportunities [2]. Two of the most common themes for earable sensing to enable HCI are head gestures and mouth gestures (including jaw, tongue, and teeth gestures) [3].

So far, head gestures have been mostly detected from inertial sensors within an earable. More specifically, accelerometer and gyroscope data [4, 5], but in-ear pressure sensing [6] has also been suggested. Jaw gestures can also be detected using in-ear pressure sensing [6]. In addition, proximity sensors [7] and piezoelectric bending sensors [8] have been researched for HCI.

For earable research, there are multiple ear-based sensing devices that offer (part of) the sensors mentioned above. The eSense [9] platform has been the most used platform for earable research. More recently, the OmniBuds [10] platform was released as a successor to eSense. The most recently released platform is the OpenEarable 2.0 [11], which offers, among others, a 9-axis Inertial Measurement Unit (IMU) located at the concha, a 3-axis IMU located at the ear canal entrance and a barometer located at the ear canal entrance.

What all of the platforms have in common is that they do not provide a magnetic sensor located at the ear canal. Currently, some platforms, such as OpenEarable 2.0, offer a 3-axis magnetic sensor located at the concha but not in the ear canal. The reason for this is that magnetic sensors have been found to be problematic due to the magnetic interference from the speakers [11]. However, these issues can be reduced with calibration methods [12] or by using piezoelectric speakers [13].

Still, magnetic sensors are only available in earables located at the concha and are mainly used for navigational purposes [12, 14]. To the best of our knowledge, current research has not explored the use of electromagnetic sensing in the ear canal for the detection of head and jaw movements for the purpose of HCI.

Recently developed earables (SOVN earbuds, Jawsaver BV, The Netherlands) use magnetic sensing of the external ear canal. The In-Ear Magnetosensing (EarMag) sensor is capable of measuring deformations of the ear canal ranging from the smallest deformations caused by heartbeat to larger deformations caused by jaw movements. More specifically, the condyle, as part of the Temporomandibular Joint (TMJ) [15], is deforming the ear canal when moving the jaw. This leads to the main question: *"How feasible is EarMag for detecting jaw and head movements to enable HCI?"*. By answering this question, the main contributions of this work are as follows:

- We present a novel dataset compiled as a part of this study, which contains the data of 17 orofacial physiotherapy related exercises collected from 21 participants.

- We show the potential of using EarMag to detect head and jaw movements to enable HCI by achieving 80% precision and 76% recall in five different movements of ten participants excluded from the training set.

The research question will be answered by focussing on orofacial physiotherapy as a proof of concept. In an interview with orofacial physiotherapist Simone Gouw [16], a list of 17 exercises was created. With the help of a data collection protocol, EarMag data was collected from 21 participants performing the 17 exercises using the EarMag-enabled earables. While investigating different signal processing techniques, the data was preprocessed, the features were extracted, and the exercises were classified using traditional machine learning techniques. To assess the feasibility of EarMag for detecting head and jaw movements, common evaluation metrics, such as precision, recall, specificity, and F1-score, were used.

The remainder of this thesis is structured as follows. In Chapter 2 the related work will be given, which will cover the current methods used for in-ear jaw and head movement recognition. In Chapter 3 the study design will be explained, which contains a detailed description of all exercises in the dataset. In Chapter 4 the methodology is described, which will outline all the steps taken from data collection to classification. After that Chapter 5 will present and analyze the findings, followed by a discussion of the findings and the limitations. After the discussion, the conclusion in Chapter 6 will summarize the findings. Lastly, Chapter 7 will suggest future research directions.

A more compact version of this thesis is available in the corresponding workshop paper [17]. The source code and data are publicly available online [18].

2

Related Work

The published articles relate to in-ear sensing methods for the detection of jaw and head gestures. Table 2.1 provides a summary of the previously published methods.

2.1. Jaw Gestures

Jaw gestures can be detected from deformations in the ear canal caused by movement of the jaw. Specifically, movement of the condyle, part of the TMJ, alters the shape of the ear canal.

CanalSense [6] is a face-related movement recognition system based on air pressure sensing with an embedded barometer. Using commercially available earbuds customized with an embedded barometer and an Random Forest (RF) model, CanalSense can detect gestures involving sliding the jaw left and right, and opening and closing of the mouth with an accuracy of 88% across 12 participants. It can also detect four stages of opening the mouth (closed, slightly open, open, fully open) with an accuracy of 80%.

In contrast, proximity sensors are proposed to measure the deformation of the ear canal caused by movement of the lower jaw [7]. More precisely, three orthogonal infrared proximity sensors attached to a custom-fitted earpiece have been implemented to classify different jaw gestures, such as opening and closing of the mouth.

Lastly, a piezoelectric bending sensor is also suggested for the detection of jaw gestures [8]. The bending sensor consists of a thin piezoelectric strip attached to a custom-fitted earpiece and was used to detect jaw gestures such as opening and closing of the mouth.

2.2. Head Gestures

While CanalSense [6] was used for the detection of four jaw gestures, it was also designed to detect head gestures that involve facing left and right, facing up and down, and tilting the head left and right. CanalSense, using pressure sensing, was able to detect the head gestures with an accuracy of 85% across 12 participants.

Head gestures can also be detected using inertial sensors. Laporte et al. [5] used the eSense earables with a built-in 3-axis accelerometer and a 3-axis gyroscope for the detection of head shaking and nodding. The paper showed the potential of using an end-to-end deep convolutional neural network by achieving a F1-score of 82% based on ten participants.

Similarly, Gashi et al. [4] also used the eSense earables with the 3-axis accelerometer and 3-axis gyroscope for the detection of head shaking and nodding. In contrast, this work collected data from 21 participants, used a hierarchical approach based on deep neural networks, and used transfer learning that involves existing human activity recognition datasets. This research reported an F1-score of 88% on the head shaking and nodding exercises.

Lastly, Groovemeter [19] detects head motion reactions to music using the eSense earables with a

Table 2.1: Overview of in-ear sensing methods for jaw and head gesture recognition.

Study/ System	Sensor Type	Target Gestures	Method/ Model	Performance
CanalSense [6]	Barometer (air pressure)	Jaw (open/close), Head	Random Forest	88% Jaw, 85% Head
IR Proximity [7]	3D IR Proximity Sensors	Jaw (open/close)	—	—
Piezo Sensor [8]	Piezoelectric bending strip	Jaw (open/close)	—	—
Laporte et al. [5]	3-axis Acc + Gyro (eSense)	Head (nodding, shaking)	Deep CNN	82% (F1)
Gashi et al. [4]	3-axis Acc + Gyro (eSense)	Head (nodding, shaking)	Hierarchical DNN + TL	88% (F1)
Groovemeter [19]	3-axis Gyroscope (eSense)	Head (music response)	LSTM (RNN)	81% (F1)

Table 2.2: Overview of in-ear sensing methods for alternative gesture recognition.

Study/ System	Sensor Type	Target Gestures	Method/ Model	Performance
CanalScan [20]	Acoustic reflections (smartphone)	Tongue-Jaw (6 types)	RF	95% precision
Barton [21]	Barometers (in-ear)	Tongue (left, right, forward)	SVM	94% accuracy
Tempo [22]	Optical sensor (earphone)	Tongue (press roof)	—	100% precision
EarSense [23]	Earphone as input transducer	Teeth (7 types)	—	90% accuracy
TeethTap [24]	Acoustic + 3-axis Gyro	Teeth (13 types)	SVM, KNN	91% accuracy
Chugh et al. [25]	3-axis Acc + Gyro (AirPods Pro)	Eye (blink, wink, brows)	GRU (RNN)	85% (F1)
BreathSign [26]	Inward-facing microphones	Breathing (3 types)	—	93% accuracy
MuteIt [27]	Dual IMU (jaw tracking)	Silent speech (100 commands)	—	95% accuracy
JawSense [28]	3-axis Acc (ear canal)	Silent speech (9 phonemes)	SVM	92% accuracy
Sahni et al. [29]	Proximity + Magnetometer	Silent speech (11 sentences)	—	91% accuracy

3-axis gyroscope. Long short-term memory, a type of recurrent neural network, was used to classify head motions collected from 30 participants. The evaluation showed an F1-score of 81% for head motion reactions with leave-one-subject-out cross-validation.

2.3. Similar Interactions

Interactions based on in-ear sensing of other types of gestures have also been scanned for inspiration purposes. Detection systems triggered by tongue gestures, teeth gestures, eye gestures, breathing gestures, and silent speech input are summarized in Table 2.2 and described in more detail below.

2.3.1. Tongue Gestures

CanalScan [20] is a tongue-jaw movement recognition system based on acoustic reflections. It uses an RF model to classify six movements involving the tongue and jaw with an average precision and recall of 95% across 20 participants. However, this method has been implemented using a smartphone and has not been tested with earables.

Barton [21], a barometer based low-power tongue movement system, was also proposed for the detection of tongue gestures in the ear. Using custom made earbuds with two build-in barometers, tongue gestures involving tongue left, right, and forward were collected from five participants. Evaluation showed that Barton with a Support Vector Machine (SVM) could classify the three tongue gestures with an accuracy of 94%, while consuming 44 times lower energy than the state-of-the-art microphone-based solutions.

Lastly, the earable Tempo [22] uses an earphone-type sensor to optically measure changes in the shape of the ear canal when the tongue is pressed against the roof of the mouth. The proposed system was evaluated on five participants, and the average precision was reported as 100% and the recall as 77%.

2.3.2. Teeth Gestures

EarSense [23] is a new approach to using earphones as a teeth activity sensor. The system was tested by 18 participants performing seven teeth gestures, including taps and slides. By changing the earphone speaker to an input transducer, EarSense is able to sense the vibrations caused by the seven teeth-related gestures with an accuracy of 90%.

TeethTap [24] recognizes discrete teeth gestures using motion and acoustic sensing on an earpiece. More specifically, by fusing acoustic and 3-axis gyroscope data, TeethTap can classify 13 discrete teeth tapping gestures from 11 participants with a real-time accuracy of 91% in a laboratory environment using

both an SVM and K-nearest-neighbour algorithm.

2.3.3. Eye Gestures

Recently, Chugh et al. [25] presented a novel framework that uses IMU sensors embedded in earables for real-time eye gesture detection. Airpods pro earphones with a 3-axis accelerometer and 3-axis gyroscope were used to collect data from 14 participants. Evaluation showed that various eye gestures, such as blinking, winking, and eyebrow movements can be classified using a gated recurrent unit with an F1-score of 85%, even when the user was in motion.

2.3.4. Breathing Gestures

BreathSign [26], an earable-based authentication system, uses inward-facing microphones on commercial earphones to capture in-ear breathing sounds for passive authentication. Fast, normal, and deep breathing gestures were performed by 35 participants and results show that BreathSign can achieve an average authentication accuracy of 93% through one breathing cycle.

2.3.5. Silent Speech Input

Mutelt [27], an ear-worn system for recognizing voice-based commands, consists of a twin-IMU setup to track jaw movements while cancelling motion artifacts caused by head and body movements. The system was validated for 20 users with diverse speech accents to recognize 100 commonly used voice commands with an average accuracy of 95% under noise-free conditions.

JawSense [28] is a wearable device that enables HCI based on unvoiced jaw movement tracking due to ear canal deformations. JawSense can classify nine phonemes with 92% accuracy based on an SVM trained on 3-axis accelerometer data of six participants.

Lastly, Sahni et al. [29] combines a 3-axis magnetometer embedded in the Google Glass device with proximity sensors embedded in a set of custom-made earpieces. The magnetometer measures movement of a small magnet attached to the tongue, and the proximity sensors measure deformations of the ear canal. The proposed system can detect 11 sentences with 91% accuracy based on the data of six participants.

In contrast to these works, this work shows how EarMag-enabled earables can be used to recognize orofacial physiotherapy-related exercises, including jaw and head gestures, and breathing and massaging exercises.

2.4. A Need for Jaw and Head Gesture Recognition

There is a growing need for reliable jaw and head gesture recognition, particularly in the context of orofacial physiotherapy. Patients recovering from conditions such as TMJ disorders perform specific exercises that involve subtle jaw or head movements. Accurate gesture detection allows for progress tracking, remote supervision, and enhanced adherence while patients continue performing these exercises at home. While broader applications exist, such as assistive technologies, silent speech interfaces, and biosignal tracking, this work focuses on physiotherapy-related use cases.

Gesture recognition modalities based on IMUs, pressure sensors, and proximity sensors face limitations in detecting subtle gestures or localized movements and are mostly targeted to either jaw gestures or head gestures. In contrast, EarMag can be used to detect both jaw and head gestures due to the orientation and location of the magnet and the magnetosensor. Especially the in-ear positioning of the magnet provides more detail and makes EarMag able to detect more subtle deformations of the ear canal.

3

Study Design

3.1. EarMag Hardware Description

In this study, a pair of earables equipped with EarMag sensors is used for tracking jaw and head movements. The use of magnetic sensors in earables is already quite common in devices available on the market, primarily due to their small size, ease of integration, and low power consumption. The in-ear portion of the earables consists of a magnetic sensor housed in the body of the earpiece and a flexible silicon tip that is embedded with a small (~2 mm) rare earth disc magnet. Figure 3.1 shows the position of the magnet and the magnetosensor. As jaw movements deform the ear canal, the magnetized silicon tip moves in conjunction with this deformation. Such movements cause a relative displacement between the magnet in the silicon tip and the sensor, resulting in variations in magnetic flux density. These variations are measured by the magnetic sensor along three spatial axes (X, Y, Z), corresponding to jaw protrusion, opening/closing, and lateral movements, respectively.

Deformations of the magnet inside the silicon tip can also be caused by smaller movements, such as heartbeat and breathing. More specifically, heartbeat from the superficial temporal artery proximal to the ear canal causes micro movements of the magnetized tip, which can be captured by the magnetosensor.

This new method of biosensing allows for high-resolution, unobtrusive tracking of jaw activity, supporting applications such as bruxism detection, sleep monitoring, eating behaviour analysis, biosignal monitoring, and hands-free interaction using jaw gestures. The biosensor compatibility and ease of integration with standard in-ear wearables (e.g. True Wireless Stereo earbuds) further enhances its practicality and potential for widespread adoption. Further details regarding jaw movement tracking and biosensing using EarMag can be found in the patent application EP4440414A1 [30].

3.2. Participant Selection

To perform a structured data collection, a protocol was designed. The data collection protocol was approved by the Human Research Ethics Committee of the TU Delft. Before the participants were recruited, it was decided that the recruitment of volunteers would be random, so there were no restrictions in age, gender or other personal information. Eventually, 21 people, 12 males and nine females, participated in this research.

3.3. Data Collection

Before starting the data collection, each participant received and signed an informed consent form. Once consent was obtained, data collection could begin.

Cleaned earables were placed in the participants' ears, and each participant was seated in front of a monitor displaying a custom graphical user interface. The first screen showed a randomly generated Universal Unique Identifier (UUID) assigned to the participant. All captured data was stored anony-

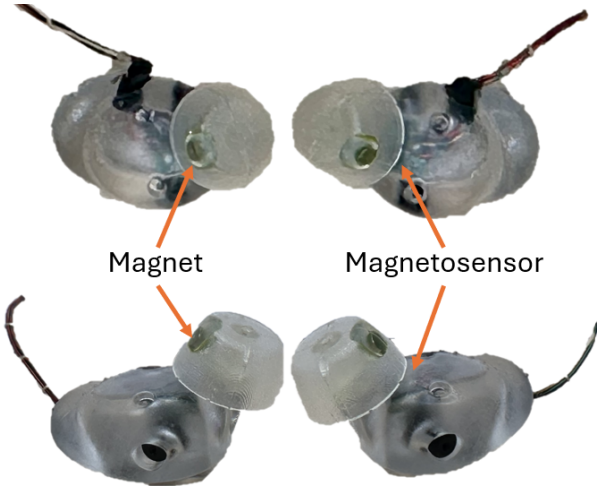


Figure 3.1: Top and side view of the EarMag-enabled earables used for the experiments. The earables are wired to a processing unit, which is not shown in the figure.

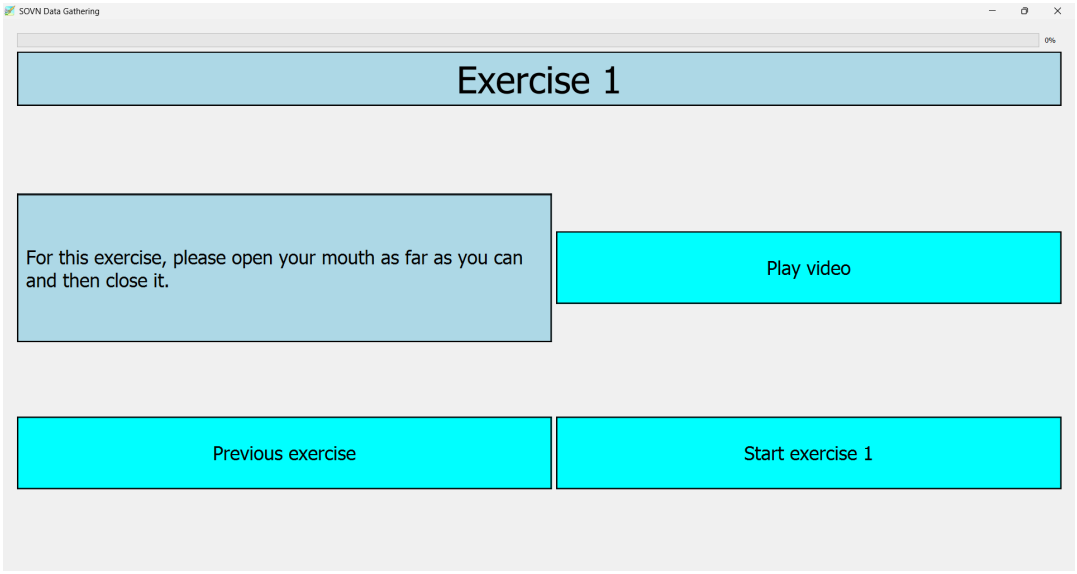


Figure 3.2: Instruction window for exercise 1, *opening-closing mouth*.

mously under this UUID, known only to the lead researcher.

Next, a welcome screen briefly introduced the study's goal, followed by an instruction screen containing both textual and video explanations of the upcoming exercise, which can be seen in Figure 3.2. It can be seen that the left section describes the exercise and at the right section a video can be viewed to see how the exercise should be performed correctly. During this time, the lead researcher monitored the session and was available to answer any questions.

Once the participant confirmed understanding, they proceeded to the exercise screen, which is shown in Figure 3.3. Most exercises consisted of three phases: performing the action within two seconds (e.g. opening the mouth), returning to a relaxed state in two seconds (e.g. closing the mouth), and pausing for one second before the next repetition. The action to be performed was highlighted in green and a short sound indicated transitions, which is particularly helpful for exercises involving head movements where the screen may not be visible. After eight iterations, the next exercise's instruction screen appeared automatically. This procedure was repeated for all 17 exercises and the specific instructions, actions, and durations can be seen in Table 3.1. The section below will describe the considered exercises in more detail.

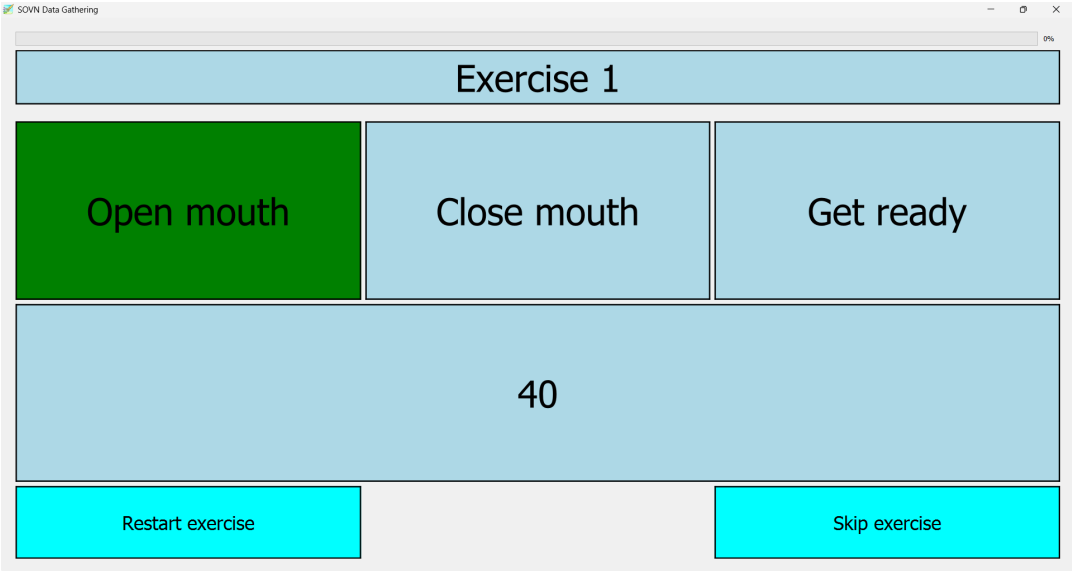


Figure 3.3: Exercise window for exercise 1, *opening-closing mouth*.

3.4. List of Considered Exercises

1. **Open close mouth:** Open the mouth to a maximum stretching position and then close it.
2. **Restricted open close mouth:** Open the mouth while positioning the tongue where you pronounce the letter 'N' and afterwards close the mouth again.
3. **Jaw left:** Move the lower jaw to the left as far as possible, where a small opening between the lips in the maximum stretching position is allowed, and then return back.
4. **Jaw right:** Move the lower jaw to the right as far as possible, where a small opening between the lips in the maximum stretching position is allowed, and then return back.
5. **Jaw forward:** Move the lower jaw forward as far as possible and then return back.
6. **Jaw backward:** Move the lower jaw backward as far as possible and then return back.
7. **Look down:** Move the chin to the chest, while not moving the upper body except for the neck, and then return the head back to the center position.
8. **Look up:** Tilt the head back and look to the ceiling, while not moving the upper body except for the neck, and then return the head back to the center position.
9. **Double chin:** Place the index finger on the chin and move the head forward without moving the upper body. Afterwards, move the head backward to create a 'double chin' without rotating the neck and upper body.
10. **Right ear to shoulder:** Move the right ear to the right shoulder, while keeping the shoulders relaxed and still, and then return the head back to the center position.
11. **Left ear to shoulder:** Move the left ear to the left shoulder, while keeping the shoulders relaxed and still, and then return the head back to the center position.
12. **Look right:** Look over the right shoulder and then return the head back to the center position.
13. **Look left:** Look over the left shoulder and then return the head back to the center position.
14. **Humming:** Try humming the letter 'N' while trying to maintain a relaxed position of the jaw, where there can be a slight opening of the mouth.
15. **Massage jaw forward:** Find the chewing muscles by placing the hands on the cheeks and clenching the teeth. Massage in this area with both index and middle fingers in a forward rotation while keeping the mouth closed.

16. **Massage jaw backward:** Find the chewing muscles by placing the hands on the cheeks and clenching the teeth. Massage in this area with both index and middle fingers in a backward rotation while keeping the mouth closed.
17. **Breathing:** Relax and breathe in through the nose, hold breath, breath out through the nose, hold breath, and repeat.

3.4.1. Selected Exercises

Out of the 17 available exercises, five exercises were selected for the classification model: *opening-closing mouth* (OC), *looking up* (LU), *looking right* (LR), *massaging jaw forward* (MJF), and *breathing* (BR). This subset was chosen because the classification complexity needed to be reduced and enough variety remains between the movement patterns. Furthermore, these exercises are relatively easy to perform considering that they will be used in a classification problem within the context of orofacial physiotherapy. The method that was used to select the five exercises will be described in Section 4.6.

Table 3.1: Data collection protocol, including a short description of the 17 exercises with the specific actions per exercise, the number of repetitions, and total duration.

Exercise label	Description	Action	Time (sec)	Repetitions	Total time (sec)
SOC	Simulation of exercise 1	Get ready Open mouth Close mouth	1 2 2	3 2 2	11
OC	Open close mouth	Get ready Open mouth Close mouth	1 2 2	9 8 8	41
ROC	Restricted open close mouth	Get ready Open mouth Close mouth	1 2 2	9 8 8	41
JL	Jaw left	Get ready Jaw left Return jaw	1 2 2	9 8 8	41
JR	Jaw right	Get ready Jaw right Return jaw	1 2 2	9 8 8	41
JF	Jaw forward	Get ready Jaw forward Return jaw	1 2 2	9 8 8	41
JB	Jaw backward	Get ready Jaw backward Return jaw	1 2 2	9 8 8	41
LD	Look down	Get ready Chin to chest Return head	1 2 2	9 8 8	41
LU	Look up	Get ready Tilt head back Return head	1 2 2	9 8 8	41
DC	Double chin	Get ready Head forward Return head	1 2 2	9 8 8	41
RES	Right ear to shoulder	Get ready Right ear to right shoulder Return head	1 2 2	9 8 8	41
LES	Left ear to shoulder	Get ready Left ear to left shoulder Return head	1 2 2	9 8 8	41
LR	Look right	Get ready Look over right shoulder Return head	1 2 2	9 8 8	41

LL	Look left	Get ready Look over left shoulder Return head	1 2 2	9 8 8	41
HU	Humming	Get ready Hum 'N' Stop	1 2 2	9 8 8	41
MJF	Massage jaw forward	Get ready Massage forward Stop Get ready	3 2 2 1	1 8 8 5	43
MJB	Massage jaw backward	Get ready Massage backward Stop Get ready	3 2 2 1	1 8 8 5	43
BR	Breathing	Get ready Breath in Hold Breath out Hold	1 2.1[1] 1[1] 2.1[1] 1[1]	1 7 7 7 7	61

[1] After every breathing cycle, the specific action durations are increased with 0.2 seconds.

4

Methodology

Figure 4.1 shows the processing pipeline of the EarMag system. The methods used to implement each part of the system will be described below.

4.1. Input Data

While the participants were performing the exercises, EarMag data was collected with a sampling rate of approximately $196\,\mathrm{Hz}$. This resulted in approximately 20 minutes of data per user from six different channels: three from the right earbud and three from the left earbud. The raw data, consisting of a timestamp and six channel values for each sample, were written to a *csv* file. Furthermore, a second file was generated for the raw annotations, consisting of the start and end timestamps, and the label of the exercise that was performed in that period of time. This resulted in 21 folders for all users, named with the created UUID tags, consisting of a raw data file and an annotation file.

4.2. Data Preparation

4.2.1. Segmentation

Segmentation of the data into shorter windows is needed, because the sensor data is recorded in the time domain. Knowing that the longest exercise duration could take up to four seconds (except for *massaging jaw* and *breathing*), window lengths larger than four seconds would only capture noise, so the following window lengths have been explored: [3.5, 3.75, 4.0] seconds. From testing, it appeared that the 3.75 second window would give the best results, so that window length was used. Another parameter for the segmentation was the overlap with the previous window, to ensure that no event was missed. From the set [50%, 65%, 75%, 80%] overlap, the 80% (three seconds) overlap gave the best results.

4.2.2. Labelling

After the segmentation was performed, each window should be assigned a label. However, this posed a slight challenge, because a window could contain multiple actions. This is because there is a one second break in between the repetitions of the same exercise and a window could contain the end and the start of two sequential iterations. To solve this, the window should be labelled as the most dominant event inside that window. To find the most dominant event, target overlap percentages were used, which describe the amount of samples from the target exercise inside the prediction window divided by the total amount of samples of the prediction window. The following target overlap percentages were tested: [50%, 60%, 70%]. When using 50% target overlap, it does capture a lot of unwanted noise, which decreases when choosing a higher target overlap. However, when the target overlap becomes too large, then short-duration movements can not be labelled any more. Consequently, for a 70% target overlap, the exercise should take at least 2.63 seconds (0.7×3.75), but some exercises can be shorter than this. For this reason, 60% target overlap was chosen.

To make the dataset balanced, 20 windows per exercise per participant were selected. For some

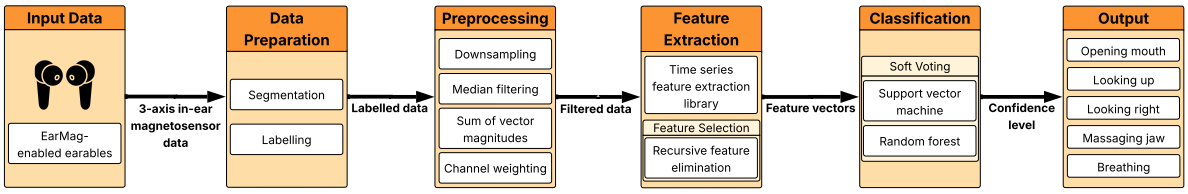


Figure 4.1: Processing pipeline of the EarMag system.

exercises, more windows were available, but in general a balanced dataset is preferred, to make the training and testing of the model more representative for the real world scenario.

4.3. Preprocessing

4.3.1. Normalization

Two types of normalization have been tested: Zero-mean normalizing (standardizing) and Min-Max normalizing. Both types of normalization can be seen in Equation 4.1,4.2 respectively. After testing, it was decided not to normalize the raw data, because it would lose important amplitude features.

$$x' = \frac{x - \mu}{\sigma}, \tag{4.1}$$

where x is the raw data, μ is the mean of the raw data and σ is the standard deviation of the raw data.

$$x' = \frac{x - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}}, \tag{4.2}$$

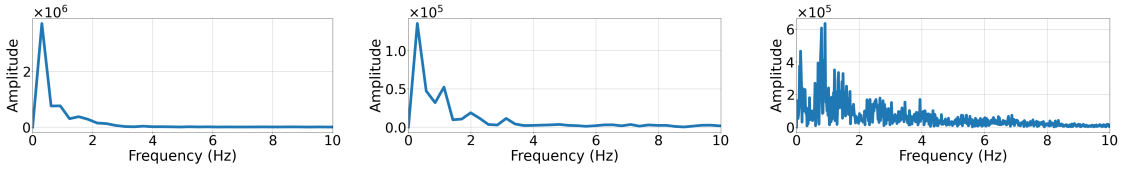
where x_{min} is the minimum value of the raw data and x_{max} is the maximum value of the raw data.

4.3.2. Downsampling

To remove high-frequency noise from the raw data, both low-pass filtering and downsampling have been tested. After testing, it was decided to only use downsampling, because it resulted in a better performance and it has some other advantages over low-pass filtering. Firstly, it reduces the data size. As a result, processing and visualizing the data will be faster, and there will be less storage requirements. Secondly, feature extraction will be simplified, because low-frequency signals offer clearer patterns, leading to more reliable features. Lastly, downsampling prevents overfitting in machine learning models. High-resolution data can cause machine learning models to overfit to noise.

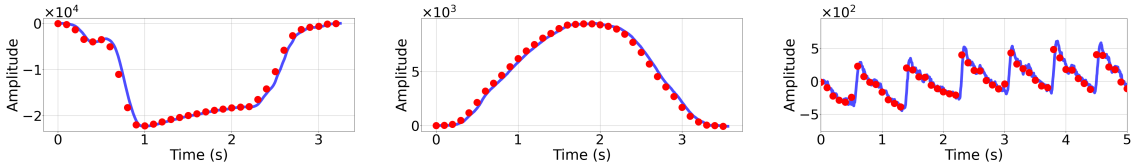
Before explaining the method that was used for downsampling, first the data should be analyzed for the highest frequency components. According to the Nyquist-Shannon theorem, the sampling frequency should be at least twice the frequency of the highest frequency component of the raw data. This theorem prevents aliasing, namely the loss of information of the original signal. Figure 4.2 shows the average frequency spectrum of a jaw exercise, head exercise, and a non-moving exercise. The first peak in Figures 4.2a, 4.2b represents the duration of the exercise ($\approx 1/0.33\,\text{Hz} = 3\,\text{sec}$). The largest peak in Figure 4.2c represents the heart rate ($\approx 0.9\,\text{Hz} \times 60\,\text{sec} = 54$ bpm). For all subfigures, it can be clearly seen that the most dominant frequencies are below $4\,\text{Hz}$. Following Nyquist-Shannon's theorem, this translates to a minimum sample frequency of $8\,\text{Hz}$, but a small safety margin was built in to end up with a sample frequency of $10\,\text{Hz}$. Compared to the original $200\,\text{Hz}$ data, this would result in a 95% decrease in data size, while it was observed that the performance did not decrease. After deciding the downsampled frequency, the method for downsampling was researched. Various methods exist, such as random sampling, decimation, and aggregated sampling. Aggregated sampling was chosen, because it is suitable for reducing the granularity of time series data while retaining the overall trend, which matches with this project. Aggregated sampling combines N data points into a single aggregated value, where $N = \frac{\text{original frequency}}{\text{downsampled frequency}} = \frac{200\,\text{Hz}}{10\,\text{Hz}} = 20$ data points. First, the original data was resampled into groups of N samples, then mean aggregation was applied to end up with a single value for every group. Lastly, linear interpolation was applied to possibly missing values.

Figure 4.3 shows the effect of downsampling for three different speed varying exercises. In general, the jaw movements have a shorter time between the relaxed state and the maximum stretched state in



(a) *Opening-closing mouth*. **(b)** *Looking right*. **(c)** *Breathing*.

Figure 4.2: The average frequency spectrum of eight repetitions of three different exercises for one user for channel one.



(a) *Opening-closing mouth*. **(b)** *Looking up*. **(c)** *Breathing*.

Figure 4.3: The average time series data of eight repetitions of three different exercises for one user for channel one. The blue line corresponds to the original $200\,\mathrm{Hz}$ data and the red dots represent the downsampled $10\,\mathrm{Hz}$ data.

comparison to the longer and smoother state changing head movements. For the opening and closing of the mouth exercise (4.3a), the downsampled data still capture the pattern of the movement, while removing the high frequency noise fluctuations. As expected, the same holds for the slower state changing looking up and return exercise (4.3b). The hardest test is visible in Figure 4.3c, where it can be seen that the downsampled data is still able to capture the pattern of the heartbeat in the breathing exercise.

4.3.3. Median Filtering

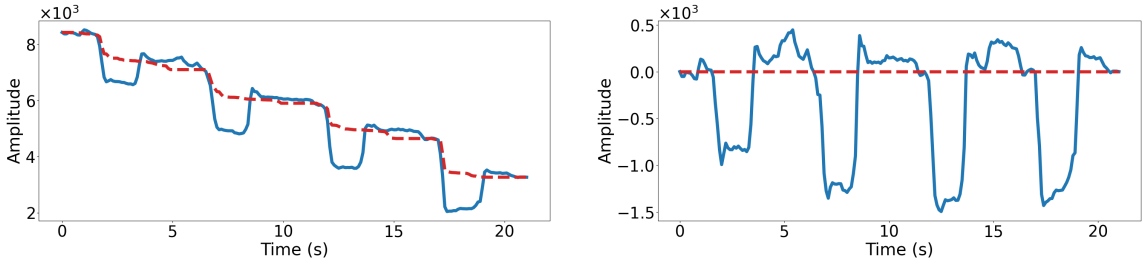
There are two main reasons for using median filtering during preprocessing. First, to center the signal. By centering the signal, the channel offsets are removed and the comparison of the individual channels across users is made more accurate. Second, to remove the sensor drift. Sensor drift that can be caused by Earth's magnetic fields or or by slight changes in the ear tip position during exercises. More specifically, the relaxed state before and after the exercise will be at different heights if the earbud position changed during an exercise. This phenomenon can be seen in Figure 4.4, specifically in 4.4a an extreme case of sensor drift for a specific user performing *opening-closing mouth* is shown. The effect of the median filter can be seen in Figure 4.4b, where the sensor drift and the sensor offsets are removed.

A median filter with a window size of six seconds was used, which is larger than the duration of a single prediction window (3.75 seconds). To handle the non-existing samples before and after the prediction window within the six second filter window, a 'nearest' mode filler was used. This method extends the prediction window samples by replicating the boundary samples (a a a a | a b c d | d d d d). The reason for choosing this method, is to make the median approximately equal to the baseline, which is equal to the start and end point of the movement. This method ensures that all exercises are centered around the baseline, which makes the comparison and processing of the data more accurate.

It should be noted that a comparable method for centering, namely mean filtering, has also been tested. However, mean filtering is more sensitive to outliers and is less effective for removing the sensor drift while keeping the overall structure the same. In combination with the better results for median filtering, mean filtering was not used.

4.3.4. Sum of Vector Magnitudes

In time series analysis, multi-channel data is commonly combined into a single representative signal, to make the processing of the data easier and faster. However, when inter-channel dynamics or directionality matter, separate channel analysis is maintained. In this case, it was assumed that the shape and magnitude of one combined signal should give sufficient information to differentiate the ex-



(a) Without filter. **(b)** With median filter.

Figure 4.4: The time series data of four repetitions of *opening-closing mouth* for one user and one channel, where sensor drift is present and removed with a median filter. The blue line represents the (filtered) data and the red line represents the median baseline.

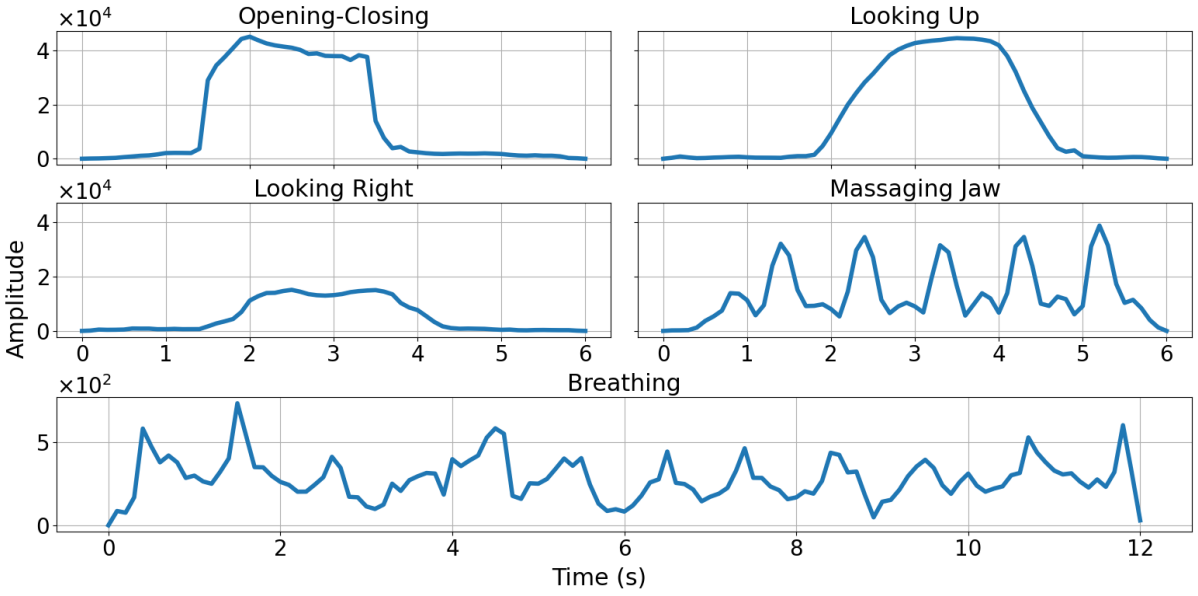


Figure 4.5: Example of filtered EarMag data of the five selected exercises for one user, where all figures show the combined SOVM signal (for definition see Section 4.3.4).

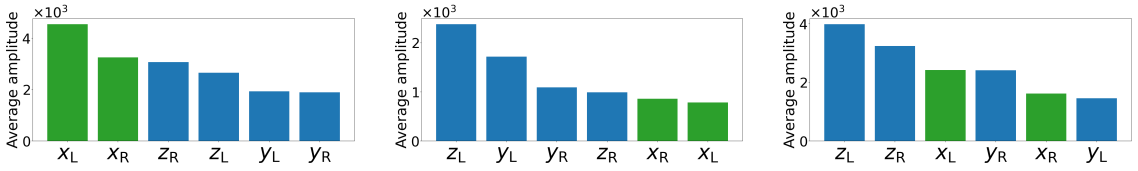
ercises. To combine the six channels into one signal, Sum Of Vector Magnitudes (SOVM) is used [31]: $\text{SOVM} = \sqrt{x_L^2 + y_L^2 + z_L^2} + \sqrt{x_R^2 + y_R^2 + z_R^2}$, where x_L, y_L and z_L represent the left earbud channels and x_R, y_R and z_R represent the right earbud channels.

Figure 4.5 shows an example of the SOVM signal for each of the five selected exercises. The SOVM signal will always be positive, because of the median filtering before calculating the SOVM signal. On the one hand, this loses some information of the specific exercise in comparison to the other exercises, but on the other hand, there will be less variability between users for the same exercise and the data reduction of 83% will have a big impact on the speed of the data processing for feature extraction and model training.

4.3.5. Channel Weighting

To improve the separability between similar exercises, channel weighting was explored as a final step in the preprocessing pipeline. The main idea was to increase the contrast between exercises by boosting channels that were strong for one gesture but weak for another.

This analysis focused on three exercises that were found to be the most similar: *opening-closing mouth*, *looking right*, and *looking up*. For each, the average amplitude across all repetitions for each channel was computed. Weighting factors were then derived to highlight distinct signal characteristics between exercises, in order to to make the combined signal more discriminative.



(a) *Looking up.* **(b)** *Looking right.* **(c)** *Opening-closing mouth.*

Figure 4.6: The average amplitude per channel of all repetitions from 21 participants for three exercises. The green bars highlight the channels that will be boosted with a factor of 1.25.

Figure 4.6 shows the average amplitudes for each individual channel for the three selected exercises. Highlighted in green are the channels that were chosen to be boosted. It can be seen in 4.6a that x_L and x_R are the biggest contributors for *looking up* while the same channels are the least important for *looking right* (4.6b). This makes them the ideal set of channels to apply channel weighting. Figure 4.6c shows that the selected channels are both in the middle and lower section for *opening-closing mouth*, but the difference with *looking up* will still be larger than it was before. The difference between *looking right* and *opening-closing mouth* will remain approximately the same, but due to the bigger differences between the other exercises, it was decided to boost channels x_L and x_R.

Based on this analysis, channels x_L and x_R were assigned a higher weight of 1.25 in the final SOVM calculation for all exercises, while the remaining channels retained a weight of 1. This resulted in better separation for the most similar exercises while it did not have a negative impact on the other exercises.

The following set of channel weights was tested: [1.1, 1.2, 1.25, 1.3, 1.4]. From this set of channel weights, 1.25 was found to give the best results when used for both channels x_L and x_R, while the remaining channels retained a weight of 1. Equation 4.3 shows the effect of channel weighting on the SOVM calculations. Higher channel weights can result in valuable information loss from other channels and smaller channel weights would not make a noticeable difference. Downscaling other channels was also tested, but did not improve the performance.

$$\text{Weighted SOVM} = \sqrt{(1.25 \times x_L)^2 + y_L^2 + z_L^2} + \sqrt{(1.25 \times x_R)^2 + y_R^2 + z_R^2}, \tag{4.3}$$

4.4. Feature Extraction and Selection

To prepare the filtered and weighted SOVM signals for classification, meaningful features need to be extracted. Afterwards, only a subset of features is selected based on feature importances.

Inspired by the feature extraction process of MedBuds [32], Time Series Feature Extraction Library (TSFEL) [33] is used. TSFEL is a Python package for efficient feature extraction from time series data. TSFEL automatically extracts 156 features that span the statistical, temporal, and spectral domains. Each domain captures different aspects of the SOVM signal. Statistical features capture the distribution of the signal, temporal features describe time-based patterns and dynamics of the signal, and spectral features analyze the frequency content of the signal.

Given the high amount of extracted features, feature selection was applied to remove redundant features and retain only the most important features. By eliminating irrelevant features, the model accuracy will be improved, and the training time will be reduced. To select the most important feature, Recursive Feature Elimination (RFE) was used. RFE selects features by recursively considering smaller and smaller sets of features, given an external estimator that assigns weights to features. First, the estimator is trained on the initial set of features and the importance of each features is obtained. Then a set of N least important features are pruned from the current set of features. That procedure is recursively repeated until K desired features is reached.

It was chosen to use a linear SVM as a base estimator, because it is relatively fast and supports feature importances represented by coefficients. To speed up the RFE process, $N = 10$ features were pruned after each round of feature elimination. For the desired number of features $K \in \{5, 10, 15, 20, 25\}$ was investigated and the results are shown in Table 4.1. It can be seen that below ten features, the loss of

Table 4.1: K selected features comparison based on mean accuracy and Standard Deviation (SD) from five-fold cross-validation. The emphasized column shows the optimal number of features.

K features		5	*10*	15	20	25
Accuracy	**Mean**	0.68	*0.74*	0.74	0.73	0.73
	SD	0.02	*0.01*	0.01	0.03	0.02

information results in a decrease in performance, while more than 15 features results also in a slight decrease in performance due to the addition of noisy, irrelevant features. The lowest amount of features with the highest mean accuracy was chosen and that is why $K = 10$ features were selected.

To take data leakage into account, it was decided to perform RFE nested inside five-fold cross-validation. Data leakage in the form of train-test contamination can happen when the feature selection is done on all available data first, and then the same data is split into train and test sets for cross-validation. In that case, the model is tested on already seen data and this results in an unrealistic representation of the real-world future data. To prevent this from happening, nested feature selection will be performed within cross-validation, and this will be discussed in more detail in Section 4.7.

After the RFE was performed, the following list of features was selected, where the first four features are selected from the temporal domain, and the remaining six features are selected from the spectral domain:

1. **Mean absolute diff:** computes mean absolute differences of the signal.
2. **Median absolute diff:** computes median of differences of the signal.
3. **Signal distance:** computes signal travelled distance.
4. **Sum absolute diff:** computes sum of absolute differences of the signal.
5. **Wavelet absolute mean 2.5 Hz:** computes Continuous Wavelet Transform (CWT) absolute mean value of the wavelet scale corresponding to $2.5\,\mathrm{Hz}$.
6. **Wavelet energy 2.5 Hz:** computes CWT energy of the wavelet scale corresponding to $2.5\,\mathrm{Hz}$.
7. **Wavelet energy 0.42 Hz:** computes CWT energy of the wavelet scale corresponding to $0.42\,\mathrm{Hz}$.
8. **Wavelet SD 2.5 Hz:** computes CWT SD value of the wavelet scale corresponding to $2.5\,\mathrm{Hz}$.
9. **Wavelet SD 0.28 Hz:** computes CWT SD value of the wavelet scale corresponding to $0.28\,\mathrm{Hz}$.
10. **Spectral spread:** measures the spread of the spectrum around its mean value.

4.5. Classification

To evaluate the classification of the five selected exercises based on the extracted features, four widely used supervised machine learning algorithms were tested: SVM, logistic regression, RF, and extreme gradient boosting. In addition to evaluating these single classifiers, ensemble learning was also explored to potentially improve robustness. Specifically, a soft voting ensemble classifier was implemented in the evaluation pipeline, which combines the output probabilities of multiple individual classifiers. In soft voting, the output of every single classifier is a probability for every class, and the final prediction is based on the average of all output probabilities across all classifiers. On the other hand, hard voting relies only on the predicted class labels. For that reason, soft voting was preferred over hard voting.

In general, there are multiple reasons why a soft voting classifier outperforms individual models. One of the biggest reasons is the combination of individual model strengths, because each model might be better at classifying different parts of the data. Table 4.2 confirms this reasoning, because all soft classifiers outperform the single classifiers. Although the differences are minimal, the accuracy did still improve. After testing ensembles up to three classifiers and comparing them with the single classifiers, it was decided to use an ensemble of SVM and RF as a soft classifier. It shared the best overall mean accuracy, but it also had the lowest SD, which indicates that the classifier is the most stable. The combination of SVM and RF works particularly well due to their diversity. SVM is a margin-based,

Table 4.2: Model comparison based on mean accuracy and SD from five-fold cross-validation. The emphasized column shows the optimal classifier.

Model		LR	RF	XGB	SVM	XGB+LR	XGB+SVM	*SVM+RF*
Accuracy	**Mean**	0.72	0.72	0.73	0.73	0.74	0.74	*0.74*
	SD	0.02	0.01	0.02	0.02	0.03	0.03	*0.01*

in this case linear, classifier that provides sharp boundaries. On the other hand, RF is a tree-based classifier that is more flexible and works well with non-linear data.

4.6. Exercise Selection

To choose five of the 17 exercises, both a clustering method and a classification score were explored. First, a clustering method will be discussed, that clusters the extracted features based on Euclidean distances. Clustering will show how similar the exercises are based on individual feature vector distances. If the feature values are close to each other, then the exercises are similar and it will be harder for a classifier to separate them.

In this case, Euclidean distance was chosen as the metric that was used for the distance calculations between the feature vectors. Other popular distance metrics are Manhattan, cosine similarity and Mahalanobis distance. The reason for choosing Euclidean distance is because it is the default for most machine learning applications and it is computationally efficient. Before the feature distances could be calculated, the feature vectors should be normalized first. Every user has a different range of motion for each exercise, so normalizing makes the feature vectors more comparable. Zero-mean normalizing (standardizing, see Equation 4.1) was used and then the feature vector distances could be calculated.

The dendrogram in Figure 4.7 shows the result of clustering based on Euclidean distances between features. Five different coloured clusters can be seen, where every cluster represents a certain set of exercises. The red cluster contains only head movements, the blue cluster only jaw movements, the orange cluster a combination of head and jaw movements, the purple cluster no motion exercises, and the brown cluster massaging exercises. The brown cluster is the farthest away from the other clusters, due to the most distinct features of the massaging exercises. The second furthest is the purple cluster with the humming and breathing exercises, which are the lowest amplitude exercises and are also very unique. Then the orange cluster contains both jaw and head movements, which have higher variability and are less distinct. The red and blue clusters show that the remaining head and jaw movements have similar features and that is why they are clustered separately. To make a selection based on the shown dendrogram, no more than one exercise from every cluster was chosen. Choosing multiple exercises from the same cluster would result in multiple misclassifications, so it was decided to choose one exercise from every cluster. Figure 4.7 shows the five selected exercises in green. However, the decision for these exercises was not only made on the feature distances. A classification score was also derived, including four different metrics that could capture the classification complexity. These metrics are Signal-to-Noise Ratio (SNR), intra-user variability, inter-user variability, and Fisher score.

The first metric that was used, is SNR. SNR was chosen to make a distinction between low and high noise exercises. In general, exercises with more noise are harder to classify than exercises with less noise. To determine which part of the signal is noise and which part of the signal contains the actual motion artifacts, frequency spectrum analysis was performed. As described in Section 4.3.2, the most dominant frequencies of all exercises are below $4\,\mathrm{Hz}$. Based on this result, the assumption was made that frequencies higher than $4\,\mathrm{Hz}$ would mostly contain noise. Then the SNR was calculated as the signal power between 0 and $4\,\mathrm{Hz}$ divided by the signal power outside the signal band. The resulting exercise ranking based on the SNR can be seen in Appendix A.1.

Next to SNR, variability also has a big impact on the classification complexity. In general, the larger the variability the larger the classification complexity. Variability occurs due to the different anatomies of participants ears, because each person has a slightly different shaped ear and ear size, which affects the sensor measurements. Furthermore, variations also occur due to the physical capabilities of a participant to perform an exercise to full extend, i.e. one person can open the mouth further than the other. Lastly, external factors, such as earbud placement or user error when performing an exercise,

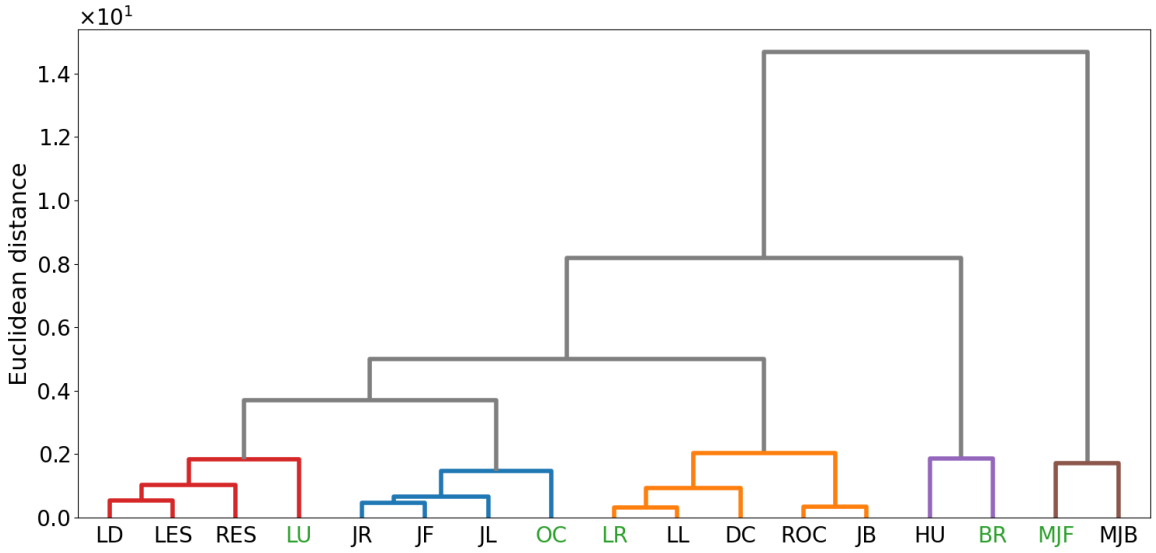


Figure 4.7: Hierarchical clustering based on Euclidean distances between the average feature vectors of all exercises. Head movements (red), jaw movements (blue), head and jaw movements (orange), low amplitude exercises (purple), and massage exercises (brown) are clustered. The green labels correspond to the five selected exercises.

are also a cause of variability. In this case, two types of variations have been defined, namely intra-user variability and inter-user variability. Intra-user variability is the variability between repetitions performed by the same user for a given exercise. Inter-user variability is the variability across different users for the same exercise.

Intra-user variability was implemented by using SD and Coefficient of Variation (CV). CV is a normalized variation of the SD by dividing the SD by the mean. Both of these metrics were averaged for all user-exercise pairs and these pairs were averaged across all users to end up with the intra-user variability for each exercise. The reason for using both SD and CV is that they give a more complete view of the real intra-user variability. SD provides an absolute measure of the variability that is present, while CV gives a relative measure, adjusting for differences in magnitude between exercise repetitions. The resulting intra-user variability for each exercise can be seen in Appendix A.2.

For inter-user variability, SD was calculated across participants to assess the spread of average performance of different users for the same exercise. However, only CV, the normalized SD, was used because of the substantial difference between the exercise means across users. The inter-user variability is visualized in Appendix A.3.

The last metric, Fisher score, describes how distinct each exercise is based on user feature data. More specifically, Equation 4.4 was used to calculate the Fisher scores. First, the mean and variance per exercise feature were computed across users. Then for all exercise pairs, the Fisher scores were computed. The averaged Fisher score for each exercise across all related exercise pairs was defined as the final Fisher score per exercise, where higher scores indicate more distinguishable exercises. The average Fisher scores per exercise can be seen in Appendix A.4.

$$\text{Fisher} = \frac{(\mu_x - \mu_y)^2}{\sigma_x^2 + \sigma_y^2}, \tag{4.4}$$

where μ_x, μ_y are the feature means for exercise x and y, and σ_x, σ_y are the feature variances for exercise x and y.

All metrics are combined in a classification score, shown in Equation 4.5. To make all metrics comparable, they were normalized to a value between zero and one. Fisher scores were assigned a weight of two, because there are two different variability metrics and feature distinctiveness is at least as important as the low variability. The variability metrics were inverted, to give low variability a high score and

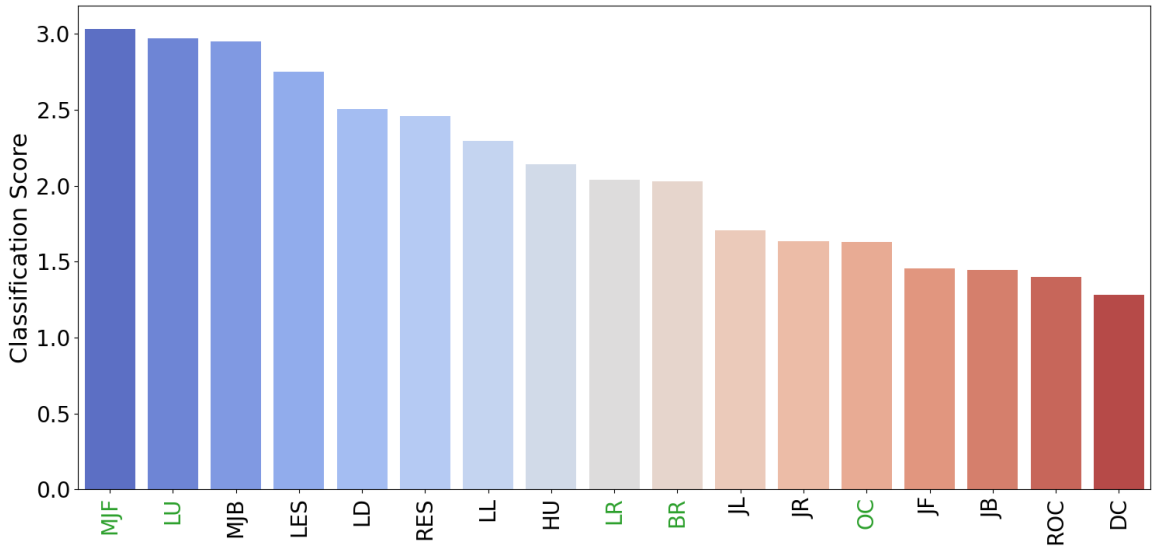


Figure 4.8: Combined classification scores for all exercises based on four metrics shown in Equation 4.5. The exercises are ordered from easiest (blue) to hardest (red) to classify. The green labels correspond to the five selected exercises.

high variability a low score. The combined classification scores can be seen in Figure 4.8. On average the jaw movements (OC, ROC, JL, JR, JF, JB) score lower than the head and other physiotherapy-related exercises. Eventually, based on three criteria, five exercises were selected. First, at least one exercise per cluster (Figure 4.7) should be present. Second, a mix of classification scores should be chosen. Third, the exercise should be trivial to perform for the purpose of HCI. The five selected exercises based on these criteria are *opening-closing mouth* (OC), *looking up* (LU), *looking right* (LR), *massage jaw forward* (MJF), and *breathing* (BR).

$$\text{Classification score} = \text{SNR}' + \text{Var}'_{\text{intra-user}} + \text{Var}'_{\text{inter-user}} + 2 \times \text{Fisher}', \tag{4.5}$$

where SNR' is the Min-Max normalized (Eq. 4.2) signal to noise ratio, $\text{Var}'_{\text{intra-user}}$ is the normalized, inverted intra-user variability, $\text{Var}'_{\text{inter-user}}$ is the normalized, inverted inter-user variability and Fisher' is the normalized Fisher score.

4.7. Evaluation Procedure

K-fold cross-validation was implemented for evaluating the models. This method ensures that the results are not biased by a certain data split and provides a more reliable image of how the model will perform in real-time. K-fold cross-validation divides the data into k (possibly) equally sized subsets. The model will then be trained on $k-1$ sets and tested on the remaining set. This process is repeated k times, where the test set is different each time. Lastly, the results are averaged for the k folds to obtain the final metric scores.

In this research, five-fold cross-validation was chosen, because it provides a balance between computational efficiency and statistical reliability. In this case, with a dataset of 21 users, splitting it into five folds ensures that each fold will include four sets of four users and one set of five users. This setup will give the model unseen users during every fold, which will prepare the model for the real-world scenario. In general, five-fold cross-validation will give a stable and generalizable estimate, which is particularly important for HCI.

Another approach that is commonly used for the evaluation of machine learning models, is leave-one-subject-out (LOSO). However, LOSO is mostly used when there is a limited number of participants, which is not the case for this project. Although LOSO has the advantage of maximizing the training set during each fold, it can become computationally expensive for larger datasets. In addition, one user in the test set might not give a good representation of the entire population. That is another reason for

choosing five-fold cross-validation.

4.8. Evaluation Metrics

To evaluate the soft voting classifier, four different metrics will be used: precision, recall, specificity, and F1-score. These metrics are widely used in supervised machine learning and for that reason they will be used in this project as well. It has to be noted that precision will be more important than recall, for the purpose of HCI. To illustrate, when a user is performing exercise A and the model predicts that exercise B was performed, then that is worse than when exercise A was performed and the model is not confident about any exercise. In the latter case, the user could just try again, but in the former case a completely different command will be given to the computer, which can lead to unexpected behaviour.

When dealing with multiple classes, a combined metric should be used to describe the overall system performance. In this case, precision, recall, specificity and F1-score will be macro-averaged for the five different classes. The macro-averaged precision, recall, specificity, and F1-score are calculated using Equation 4.6, where precision $= \frac{\mathrm{TP}}{\mathrm{TP+FP}}$, recall $= \frac{\mathrm{TP}}{\mathrm{TP+FN}}$, specificity $= \frac{\mathrm{TN}}{\mathrm{TN+FP}}$, and F1-score $= 2 \times \frac{\mathrm{precision} \times \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$.

$$\text{Macro-average} = \frac{1}{5} \sum_{\text{class i}=1}^{5} \text{Metric}_{\text{class i}}, \tag{4.6}$$

where Metric is precision, recall, specificity or F1-score and class 1 to 5 represent the selected exercises.

Weighted average is another option in combining the individual class metrics, but this is only useful when dealing with imbalanced datasets. In this work, during data segmentation and labelling, the dataset was made balanced, which means that every class has the same amount of labelled windows. This is particularly useful for macro-averaging, because each class is given the same weight. It has to be noted that the macro-averaged recall in a balanced dataset is the same as accuracy, which is why accuracy is not used as evaluation metric and will not be reported in the results chapter.

4.9. Validation

To assess the applicability and robustness of the trained ensemble model in real-time, a validation study was performed. The goal of the validation was to determine whether the trained model could correctly classify the selected exercises in both controlled and in-the-wild settings.

First, the in-the-wild validation was conducted. In this validation, participants were instructed to perform the selected exercises with four repetitions. However, participants were not told how fast to execute the exercises and how long to wait between repetitions. The goal of this approach was to replicate real-life HCI, where users naturally perform certain commands.

Second, the controlled validation was done. The same participants were asked to perform the same selected exercises, but this time in the same structured and timed setting as during data collection for the training set. This resulted in a direct comparison with the performance of the model under consistent conditions for all participants.

To perform validation, a group of ten participants who were not present during training and testing of the model was recruited. This number of participants was chosen based on the size of the initial training and test dataset, which consisted of 21 users. Choosing around 50% of this set for validation was considered sufficient to evaluate generalizability.

The exercises that were used during validation are a mix of nine different exercises. The first five exercises were the selected exercises that the model was trained on, which represent the set of commands that is intended for real-time HCI. The remaining four exercises were selected as reference exercises. The idea was to include these exercises as negative commands to evaluate the ability of the model to differentiate target commands to similar, non-target exercises. Two jaw movements, *moving jaw left* and *moving jaw forward*, were chosen and two head movements, *looking down* and *moving right ear to right shoulder*, were chosen.

4.10. Feature Weighting

To address the substantial variability in the performance of the exercises between users, individual feature weighting was applied. The goal of individual feature tuning was to optimize the performance of the model for each individual while using the same general model to classify the exercises for all users. The idea was to use individual feature tuning as a calibration phase before real-time classification. The future user will then first be asked to perform a few repetitions of the five selected exercises, and then the real-time command recognition will be personalized for that specific user.

To find the optimal feature weights for each individual, Optuna, an open-source hyperparameter optimization framework, was used. Optuna performs efficient parameter searches by intelligently sampling hyperparameter combinations and pruning unpromising trials early. To keep the search compact, the range of feature weights was restricted to $[0.75, 1.25]$. Furthermore, the number of trials needed to find the optimal parameters needs to be as low as possible due to time constraints, but still high enough to achieve the optimal performance. From preliminary testing, it was found that 100 trials were sufficient to achieve the highest performance in the least amount of time. During each trial, the objective was to maximize both the precision and the accuracy of the predictions. This resulted in the objective function shown in Equation 4.7. Accuracy was chosen, because it is a general indicator for the whole systems performance. In addition, the precision of the model predictions was chosen, because one of the most important things in HCI is to keep the false positives as low as possible while having as many true positives as possible. Due to the importance of precision, it was decided to give a higher weight of two to the precision metric. It has to be noted that the models confidence for the true label was also considered, because predictions will only be recognized as a true positive if the confidences are above a certain threshold. Still, precision and accuracy were considered more important and resulted in the best outcome for the purpose of HCI.

$$\text{objective} = \frac{\overline{\text{accuracy}} + 2 \times \overline{\text{precision}}}{3}, \tag{4.7}$$

where $\overline{\text{precision}}$ is the mean precision, and $\overline{\text{accuracy}}$ is the mean accuracy.

To perform the hyperparameter search in the least amount of time, the amount of data to be used for tuning should be minimized. However, if the hyperparameter search is not supplied with a representable amount of data, then the resulting feature weights will not be accurate. It was found that tuning on five windows per exercise, resulted in the most optimal feature weights in the least amount of windows. The comparison with other amounts of tuning windows is discussed in the next chapter and shown in Table 5.4.

To prevent data leakage during the process of finding the most optimal feature weights, the windows used for the hyperparameter search were excluded from the test set. To illustrate, if n is the number of windows used for tuning, then the remaining $20 - n$ windows are used for testing. This process simulates the real-world scenario, where the data captured during calibration is separated from the data captured in real-time.

5

Results and Discussion

This chapter will describe and discuss the results of five-fold cross-validation during training and after validation for the five selected exercises. For validation, both the results for controlled and in-the-wild environments will be discussed. Furthermore, the impact of individual feature weighting will be described.

5.1. Training and Testing

Table 5.1 shows the macro-averaged results for the chosen metrics after the cross-validation was performed. It can be seen that the standard deviations are low across all metrics, which means that the metric results are approximately the same score for all five folds and there are no substantial outliers.

To assess how the individual exercises contributed to the final results, a confusion matrix was derived and can be seen in Figure 5.1a. The figure shows the averaged number of predictions versus the averaged number of true movements. Each row adds up to approximately 84 predicted windows, which is the result of having four folds with four users in the test set and one fold with five users in the test set. For every exercise, 20 windows were selected per user, so $\frac{4\times4\times20+1\times5\times20}{5} = 84$ windows. The true positives can be seen in the diagonal of the matrix and due to the distinct features of *massaging jaw* and *breathing*, they score relatively higher than the other exercises.

Another observation that can be made from Figure 5.1a is that *looking right* (LR) is causing the most misclassifications while also having the least amount of true positives. From the head movements it was known that *looking right* was the hardest one to classify and that corresponds to the confusion matrix shown here. To show the difference when *looking right* was left out from the train and test set, another confusion matrix was generated with the four remaining exercises and that result can be seen in Figure 5.1b. The difference in the number of true positives can be clearly seen on the diagonal, and most misclassifications are now caused by both *opening-closing mouth* (OC) and *looking up* (LU).

To further illustrate the difference between a system with and without *looking right*, classification results for the set without *looking right* are shown in Table 5.2. It can be seen that there is a relatively high difference in performance compared to the five selected labels, with an increase of approximately 10% for almost every metric when removing *looking right* from the train and test set. To refer back to the criteria for choosing the five selected exercises, the idea was to choose at least one exercise from every cluster presented in Figure 4.7, which resulted in five exercises. However, giving the relatively high misclassification rates when using the five selected exercises compared to four selected exercises, the option for choosing four instead of five exercises was also considered for future real-time implementations. Eventually, there is a trade-off between the number of commands and the performance of the system, meaning the more commands the worse the performance.

Table 5.1: Macro-averaged results from five-fold cross-validation based on the four different metrics for the soft voting classifier.

	Precision	Recall	Specificity	F1-score
Mean	0.75	0.74	0.93	0.74
SD	0.02	0.01	0.00	0.02

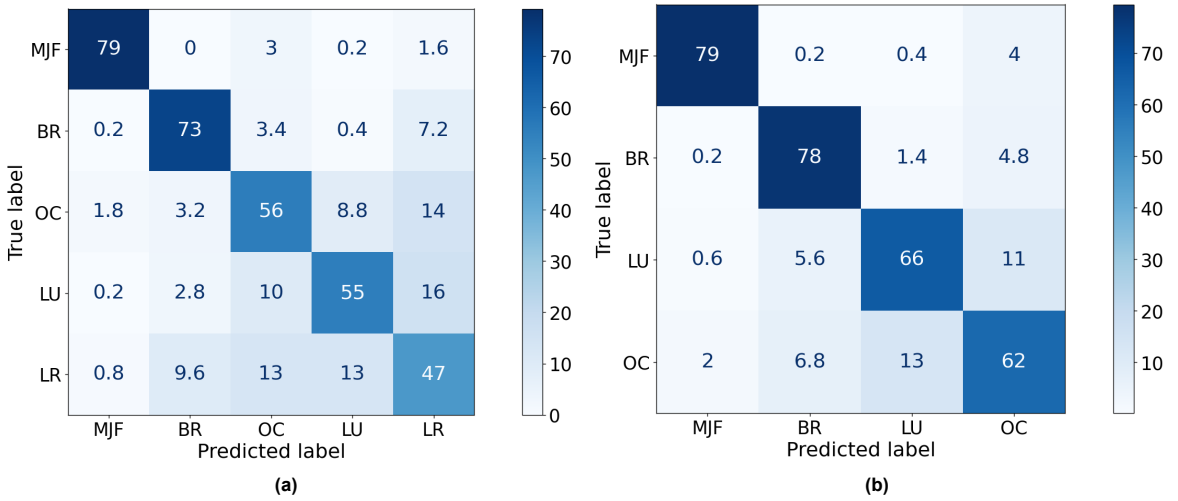


Figure 5.1: Average confusion matrix from five-fold cross-validation for the soft voting classifier, where (a) shows the results for the five selected exercises and (b) without *looking right* (LR).

5.2. Controlled Validation

We need to understand how the model would perform for unknown users. To find the optimal confidence threshold after the feature weighting process for the ten users, different confidence thresholds have been tested. The comparison of the effect of the different thresholds can be seen in Table 5.3. It can be seen that there is a general trend visible in recall and specificity, where the recall is decreasing and specificity is increasing when increasing the threshold. A higher threshold increases the number of false negatives and decreases the true positives for uncertain model predictions, which decreases the recall. In addition, the false positives decrease when increasing the threshold, which increases the specificity. A similar reasoning can be given for the increase of the precision due to the lower number of false positives at higher thresholds. However, when the threshold becomes higher than 0.5, a drop in precision is visible due to the decrease in true positives.

Based on the results presented in Table 5.3, no clear choice could be made for selecting the most optimal confidence threshold. The precision should be as high as possible, but the recall should still be in the same range. In this case, a threshold of 0.5 looks the most optimal with a precision of 0.79, but the recall of 0.68 is less optimal.

When zooming in on how each user is contributing to the average metric results, a respectively large variation was observed between the different users. Figure 5.2 shows the individual user results for every metric, where the average is given by the red dotted line. It can be clearly seen that there is a substantial variance between the individual user results for precision, recall, and F1-score. Possible reasons for these variations are differences in the anatomy of the ears of the participants or differences in the performance of the exercises, which are described in more detail in Section 4.6. The variance shown in the figure was one of the main reasons to investigate individual feature weighting, which could potentially decrease the variance between the user results and improve the overall performance of the system.

5.2.1. Feature Weighting

This section presents the results when feature weighting was applied to each individual user before the predictions were made. First, the number of windows to be used for tuning was derived.

Before applying the Optuna hyperparameter search, the validation data was split into tuning and testing

Table 5.2: Mean results from five-fold cross-validation based on the four different metrics without *looking right*.

	Precision	Recall	Specificity	F1-score
Mean	0.85	0.85	0.95	0.85
SD	0.02	0.02	0.01	0.02

Table 5.3: Results for the ten user validation with different confidence thresholds.

Confidence threshold	Precision	Recall	Specificity	F1-score
0.2	74.4	75.9	94.0	73.6
0.3	74.6	75.7	94.0	73.6
0.4	75.2	73.3	94.7	72.6
0.5	78.6	67.7	96.7	70.5
0.6	77.7	59.2	98.1	64.2
0.7	73.1	50.2	99.3	56.1

sets. To find the optimal number of tuning windows per exercise per user, different amounts of windows have been tested and reported in Table 5.4. The table shows the results after tuning for a confidence threshold of 0.2, where the optimal number of windows is highlighted. It can be seen that for five tuning windows the precision is one of the highest, while the recall is also in the same range. Using eight tuning windows does improve the performance, but that would result in a longer calibration for the real-time implementation. It was decided that five windows was the optimal balance between calibration time and performance.

To find the optimal confidence threshold after the feature weighting process with tuning on five windows, different confidence thresholds have been tested and shown in Table 5.5. The same trend as the non-weighted features is visible, where the recall and F1-score decrease, the specificity increases, and the precision first increases and then decreases, when the threshold increases. With a precision of 80.3% and a recall of 75.5%, 0.4 is the optimal confidence threshold and is highlighted in the table. It can be seen that the threshold of 0.4 has one of the highest precisions and the recall is in the same range, so that made it the best choice.

To further investigate the effect of the tuning on five windows and the confidence threshold of 0.4, the individual user results for each metric are shown in Figure 5.3. In comparison to the non-weighted feature results shown in Figure 5.2, almost all individual user results did improve and the results are closer to the average. Unfortunately, the feature weighting did not have any impact on the performance of the second user from the left, which is still an outlier. However, given the improvements of the other nine users, the implementation of feature weighting is shown to be successful. To summarize, the average precision increased with 5.1%, the recall with 2.2%, the specificity with 0.4%, and the F1-score with 2.6%.

After investigating the effect of feature weighting on the individual users, it was decided to also look into the individual contributions of the five selected exercises to the average metric results. Figure 5.4a shows the performance of the selected exercises, where each coloured bar represents a specific exercise given in the legend. Compared to the confusion matrix shown in Figure 5.1a, the same conclusions can be drawn here. *Breathing* and *massaging jaw* are still the best performers, and *looking up* and *looking right* are still the worst performers. With an average precision and recall of around 60%, *looking right* is the hardest exercise to classify for the ensemble model. Based on these results, it should be noted that the average precision and recall of *looking right* are considered too low for a stable and user-friendly HCI system.

After retraining the model on the four selected exercises without *looking right*, the improvements are relatively large and can be seen in Figure 5.4b and in Table 5.6 with the feature weighting and the confidence threshold of 0.4. The figure shows that the variance between the individual exercises has decreased substantially. The more precise differences are shown in the table where it can be seen that the differences in the average metric results can be up to 15%. With all metric averages being above 90% for the four selected exercises, using four instead of five labels results in stable and robust model predictions.

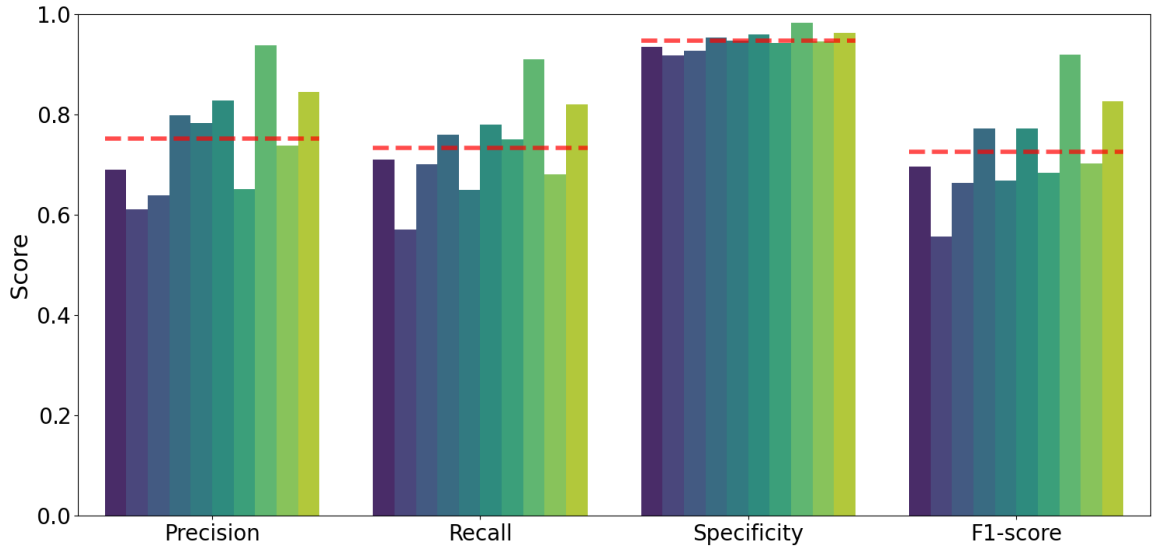


Figure 5.2: Results for each individual user in the validation set with a confidence threshold of 0.4. Each coloured bar represents a specific user and the average metric results are given with the red dotted lines.

Table 5.4: Results for the ten user validation with tuning on different number of windows for confidence threshold of 0.2. The emphasized row shows the number of tuning windows with the best overall result.

Tuning windows	Precision	Recall	Specificity	F1-score
2	75.6	76.8	94.2	74.6
4	76.4	77.4	94.3	75.2
5	*78.9*	*77.5*	*94.4*	*75.6*
6	78.8	77.9	94.5	76.0
8	80.4	78.7	94.7	76.7

5.2.2. Reference Exercises

So far, the four reference exercises that are similar to the head and jaw movements of the five selected exercises have been neglected, because the model was not trained on those exercises. Nevertheless, similar exercises should be predicted as true negatives, which is measured by specificity. To see how the model will respond to the four similar exercises, the average specificity across all ten users in the validation set will be presented.

First, model predictions were performed on only the four reference exercises in the validation dataset. Then the specificity was calculated as the number of predicted labels with a confidence less than a certain threshold (i.e. true negatives) divided by the total number of predictions. The results are shown in Table 5.7, which shows that the reference exercises are very similar to the selected exercises. It can be seen that even for a relatively high threshold of 0.7, the specificity is only 65.7%. This means that the model is highly confident about the selected commands, even though a similar reference command was given.

To further illustrate how similar the reference exercises are to the selected exercises, a heatmap was generated, which is shown in Figure 5.5. The figure shows the true reference exercises on the y-axis and the predicted selected exercises on the x-axis for all confidences above 0.4. It can be seen that the jaw movements (JL and JF) are mostly classified as *opening-closing mouth* (OC), and the head movements (LD and RES) are classified as *looking up* (LU) and *looking right* (LR). It also can be seen that *massaging jaw* (MJF) and *breathing* (BR) are distinct, which is as expected when compared to two jaw and two head movements.

Next to the number of misclassifications above the confidence threshold of 0.4, the mean confidence of these misclassifications provides a more complete view of the similarity between the exercises. Fig-

Table 5.5: Results for the ten user validation with tuning on five windows and different confidence thresholds. The highlighted row shows the results for the chosen threshold.

Confidence threshold	Precision	Recall	Specificity	F1-score
0.2	78.6	77.5	94.4	75.6
0.3	78.6	76.9	94.4	75.3
0.4	*80.3*	*75.5*	*95.1*	*75.2*
0.5	80.6	68.3	97.0	71.3
0.6	79.4	56.7	98.8	62.4

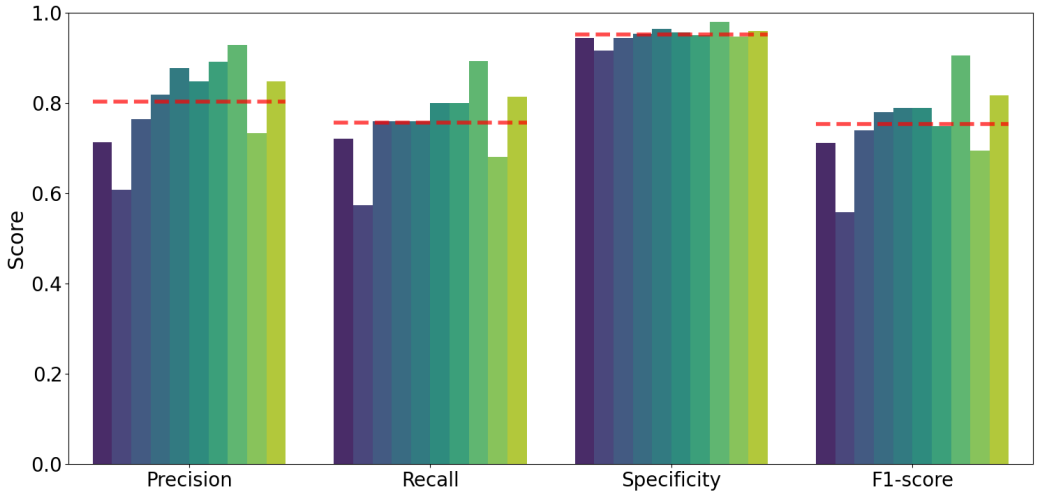


Figure 5.3: Results for each individual user in the validation set after tuning on five windows and 0.4 confidence threshold. Each coloured bar represents a user and the average metric results are given with the red dotted lines.

ure 5.6 shows that the model is the most confident about LU when the reference head movements were performed, indicating that these are the most similar. For the other head movement, LR, the mean confidences are relatively low when the reference head movements were performed. The jaw movements are slightly less similar in terms of mean confidence for OC compared to LU, but the model still thinks that the reference jaw movements are close to *opening-closing mouth*.

Lastly, the most noticeable in the heatmap are the high mean confidences for the few misclassifications of MJF. The reason for these high mean confidences is mostly the incorrect execution of the exercises. During the data collection process, some participants started performing the exercise too early, then moved back to a relaxed position, and then started the actual exercise according to the indicated instructions on the screen. In the data, this behaviour looks similar to the massaging exercise, where there are multiple consecutive peaks and valleys. This indicates that the model would only correctly classify the performed exercise, if the user executes the full movement from start to finish without interruptions.

5.3. In-the-Wild Validation

In this section, the findings will be discussed for the in-the-wild validation. In general, it was found that the results were inconsistent among the ten users in the validation set and would not give any valuable information about the performance of the model, which is why the metric results will not be discussed here. However, the in-the-wild validation did provide some valuable insight into the limitations of the current system, and these will be discussed below.

First, it was found that the duration of the execution of the exercise, has an impact on the model performance. During segmentation and labelling of the data, it was stated that the windows were only labelled if more than 60% of the windows were samples from a certain exercise. This means that the model would not be familiar with exercise durations shorter than $0.6 \times 3.75 = 2.25$ seconds. In general,

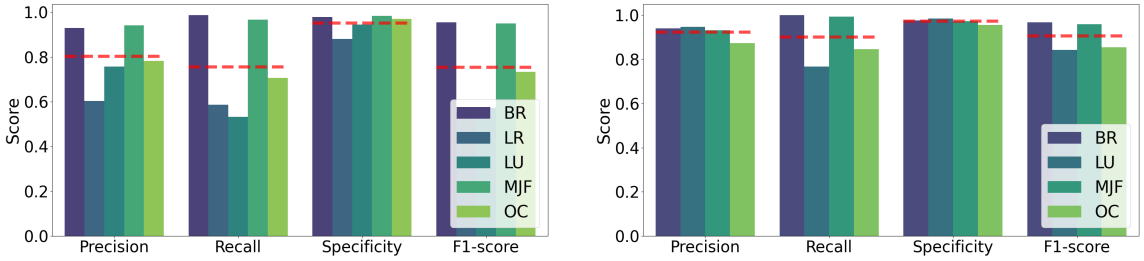


Figure 5.4: Macro-averaged metric results per label across all users in the validation set with tuning on five windows and confidence threshold of 0.4, where (a) shows the results for the five selected exercises and (b) without *looking right*.

Table 5.6: Macro-averaged results with tuning on five windows and confidence threshold of 0.4 for the five selected exercises and without *looking right*.

Number of labels	Precision	Recall	Specificity	F1-score
5	80.3	75.5	95.1	75.2
4	92.3	90.2	97.2	90.6

jaw movements can be performed faster than head movements, which was also observed for *opening-closing mouth* during validation.
Instead of fast-executed exercises, slow-executed exercises can also have a negative impact on the model performance. In particular, movements that take longer than the window duration of 3.75 seconds, will be harder or even impossible to predict with the current setup. This is not a problem for *massaging jaw* and *breathing* where the patterns stay the same throughout the exercise, but for jaw and head movements the prediction window would only contain the beginning or the end of the movement or something in between. During the validation, it was observed that some participants performed the head movements longer than the window duration of 3.75 seconds.

Second, it was found that the time between exercises also has a big impact on the model performance. If the time between repetitions of the same exercise or different exercises is shorter than the window duration, then there will be windows with multiple events within the same window. The model is not familiar with these situations, which results in multiple misclassifications. When the windows contain the beginning and the end of two different exercise repetitions, then this pattern would be similar to the massaging exercise, and therefore the windows would be mostly classified as *massaging jaw*.

Lastly, similar to the second limitation, windows that contain transitions from a relaxed state to the beginning of an exercise or from the end of an exercise to a relaxed state also cause problems. Windows that contain less than 60% of samples from an exercise will be harder to predict. When reflecting on the choice of using a sliding window approach of 80% overlap or a 0.75 second hop, the advantage is that there is always a window that fits the (complete) exercise. However, the downside of choosing a high overlap percentage is that there are more transition windows that cause misclassifications. Possible solutions for the discussed limitations will be given in the future research chapter.

Table 5.7: Specificity results for the four reference exercises with different confidence thresholds.

Confidence threshold	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Specificity	0.0	0.4	9.4	30.0	51.9	65.7	79.9	93.1

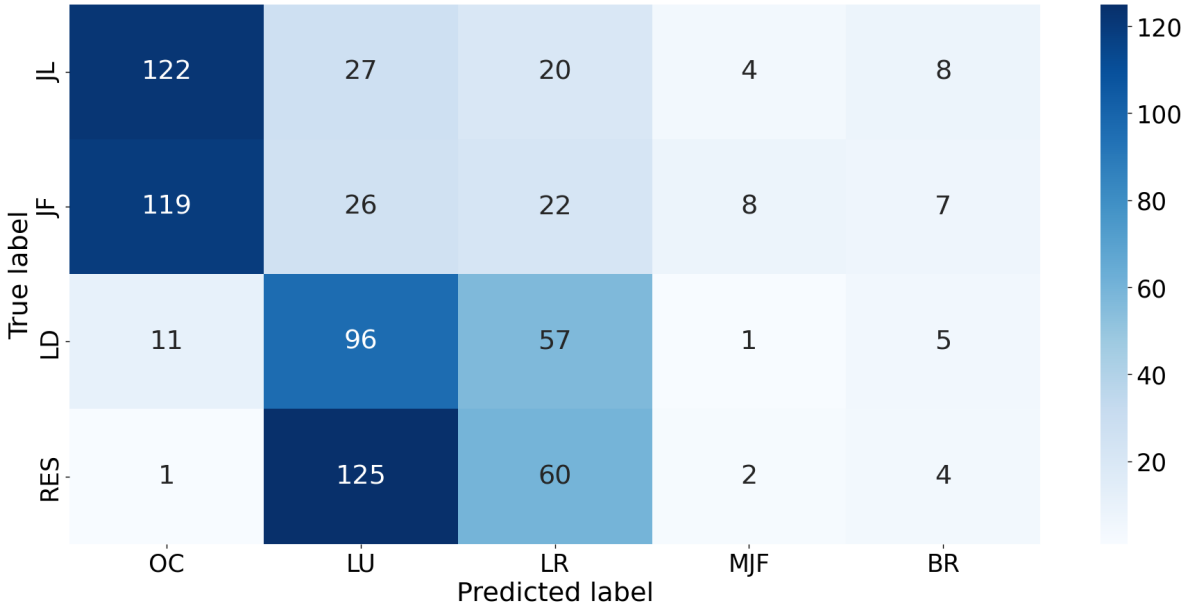


Figure 5.5: Validation heatmap of the number of predictions for the four reference exercises with a confidence above 0.4.

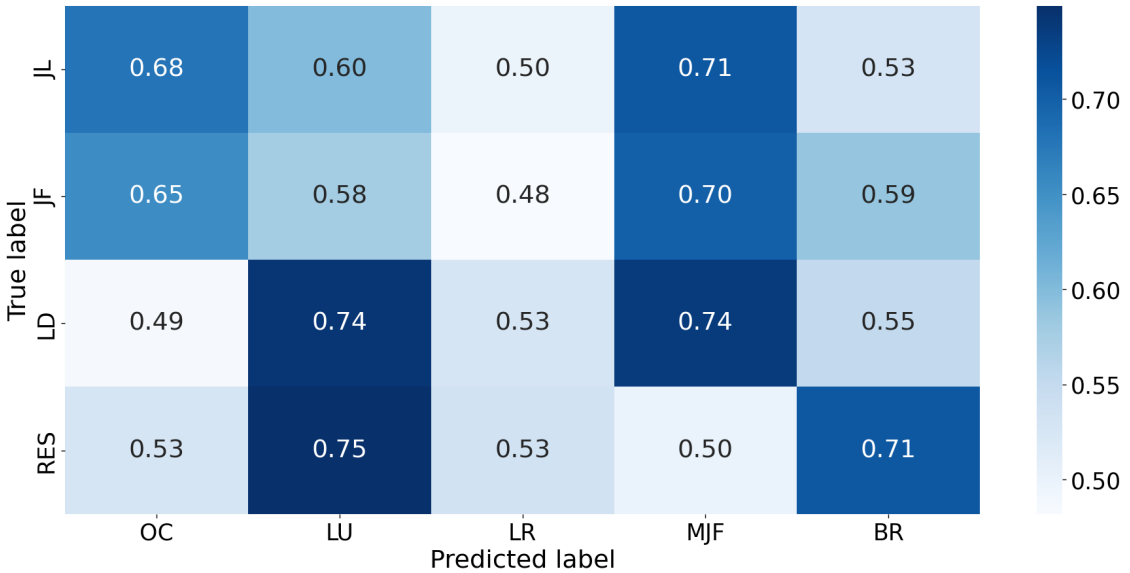


Figure 5.6: Validation heatmap of the mean confidences of the predictions for the four reference exercises with a confidence above 0.4.

6

Conclusion

The purpose of this thesis was to analyze whether the EarMag sensor data is feasible for the detection of jaw and head movements to enable HCI. To achieve this goal, EarMag data was collected from 21 participants performing 17 orofacial physiotherapy related exercises using EarMag-enabled earables. Then the collected data was prepared for preprocessing by segmenting the data into windows of 3.75 seconds with 80% overlap, and labelling the windows as the most dominant event, which is defined if at least 60% of the window contains the target event. Four steps in preprocessing were taken: downsampling from $200\,\mathrm{Hz}$ to $10\,\mathrm{Hz}$, median filtering on six second windows, calculating the SOVM, and weighting the left x and right x channels by 1.25. Then the features were extracted using the TSFEL and ten features from the temporal and spectral domain were selected using RFE. The selected features are then used for classification where a soft voting model is used consisting of a SVM and RF classifier. Lastly, the model was validated in a controlled environment using EarMag data collected from ten users excluded from the training and testing set, where individual feature weighting was applied to decrease the variance between the users and improve the overall performance.

With an average precision of 80%, recall of 76%, specificity of 95%, and F1-score of 75% at a confidence threshold of 0.4, the designed system based on the EarMag data is found to be feasible for the detection of five different exercises: *opening-closing mouth*, *looking up*, *looking right*, *massaging jaw*, and *breathing*. The hardest exercise to classify was *looking right*, with an average precision and recall around 60%. When this exercise was neglected, the performance was boosted to a precision of 92%, recall of 90%, specificity of 97%, and F1-score of 91%.

The second part of the question was to determine how feasible the system would be for HCI. After predicting the given commands in real-time during in-the-wild validation, it was found that the current setup works best when the gesture has a duration of 2-4 seconds with an interval of 3.75 seconds between gestures. Further research is needed to determine the most optimal system configuration for using EarMag for gesture detection in HCI.

7

Future Research

In this final chapter, future research is discussed based on the limitations of the current system. The following steps could be taken to improve the current model and make it feasible for HCI.

- **Traditional Learning.** This project was limited to traditional machine learning methods, but deep learning algorithms have recently been shown to outperform traditional machine learning methods in the time series classification domain. This thesis has shown that the current model is not capable of distinguishing different jaw movements and is partly capable of distinguishing head movements. Deep learning methods might learn more complex features that would be able to differentiate similar jaw or head movements and improve the performance of the entire system.
- **Short Command Durations.** The current command duration is limited to approximately two to four seconds, but a possible solution might be to change the segmentation and labelling process. Instead of labelling the entire exercise, labelling both the beginning and the end of the exercise separately would be more precise. Then the window length can be reduced to two seconds, which would be the maximum duration of starting or ending an exercise. This would remove the time constraint for the command duration, because the model would just wait for the ending label of the exercise, when the model was confident about the start of an exercise.
- **Delay Caused by Prediction Window.** The limitation of waiting at least one prediction window to give a new command can be solved by a cool-down period. The cool-down period would start when a sequence of the beginning and end of the exercise is detected. With a cool-down period equal to the duration of the prediction window, the predictions on the transition windows in the cool-down period will be neglected.
- **Lack of Comparison to Other Methods.** Another limitation is the lack of comparison to other in-ear methods of gesture detection. While our approach demonstrates promising results, it does not compare its efficacy and accuracy against existing methods and gestures used in previous HCI studies. Such comparisons are crucial as they provide a benchmark for evaluating the performance and usability of new technologies. Additionally, future research should compare power consumption and data & algorithm processing requirements for each method, as these factors are crucial considerations in embedded systems.
- **Sensor Fusion.** Lastly, future research could include the exploration of sensor fusion. Only EarMag data provided by a 3-axis magnetic sensor has been considered in this project, but sensor fusion with a 3-axis accelerometer or gyroscope may improve the detection accuracy of the commands or increase the range of commands that can be detected by the earables.

References

[1] Kaushik Ranade, Tanmay Khule, and Riddhi More. *Object Recognition in Human Computer Interaction:- A Comparative Analysis*. 2024. DOI: 10.48550/arXiv.2411.04263.

[2] Thomas Sepanosian and Ozlem Durmaz Incel. “Boxing Gesture Recognition in Real-Time using Earable IMUs”. In: *Companion of the 2024 on ACM International Joint Conference on Pervasive and Ubiquitous Computing*. UbiComp ’24. Melbourne VIC, Australia, 2024, pp. 673–678. DOI: 10.1145/3675094.3680524.

[3] Tobias Röddiger, Christopher Clarke, Paula Breitling, Tim Schneegans, Haibin Zhao, Hans Gellersen, and Michael Beigl. “Sensing with Earables: A Systematic Literature Review and Taxonomy of Phenomena”. In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6.3 (Sept. 2022). DOI: 10.1145/3550314.

[4] Shkurta Gashi, Aaqib Saeed, Alessandra Vicini, Elena Di Lascio, and Silvia Santini. “Hierarchical Classification and Transfer Learning to Recognize Head Gestures and Facial Expressions Using Earbuds”. In: *Proceedings of the 2021 International Conference on Multimodal Interaction*. ICMI ’21. Montréal, QC, Canada, Oct. 2021, pp. 168–176. DOI: 10.1145/3462244.3479921.

[5] Matias Laporte, Preety Baglat, Shkurta Gashi, Martin Gjoreski, Silvia Santini, and Marc Langheinrich. “Detecting Verbal and Non-Verbal Gestures Using Earables”. In: *Adjunct Proceedings of the 2021 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2021 ACM International Symposium on Wearable Computers*. UbiComp/ISWC ’21 Adjunct. Virtual, USA, 2021, pp. 165–170. DOI: 10.1145/3460418.3479322.

[6] Toshiyuki Ando, Yuki Kubo, Buntarou Shizuki, and Shin Takahashi. “CanalSense: Face-Related Movement Recognition System based on Sensing Air Pressure in Ear Canals”. In: *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*. UIST ’17. Québec City, QC, Canada, 2017, pp. 679–689. DOI: 10.1145/3126594.3126649.

[7] Abdelkareem Bedri, David Byrd, Peter Presti, Himanshu Sahni, Zehua Gue, and Thad Starner. “Stick it in your ear: building an in-ear jaw movement sensor”. In: *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers*. UbiComp/ISWC’15 Adjunct. Osaka, Japan, 2015, pp. 1333–1338. DOI: 10.1145/2800835.2807933.

[8] Johan Carioli, Aidin Delnavaz, Ricardo J. Zednik, and Jérémie Voix. “Piezoelectric Earcanal Bending Sensor”. In: *IEEE Sensors Journal* 18.5 (2018), pp. 2060–2067. DOI: 10.1109/JSEN.2017.2783299.

[9] Fahim Kawsar, Chulhong Min, Akhil Mathur, and Alessandro Montanari. “Earables for Personal-Scale Behavior Analytics”. In: *IEEE Pervasive Computing* 17 (July 2018), pp. 83–89. DOI: 10.1109/MPRV.2018.03367740.

[10] Alessandro Montanari, Ashok Thangarajan, Khaldoon Al-Naimi, Andrea Ferlini, Yang Liu, Ananta Narayanan Balaji, and Fahim Kawsar. *OmniBuds: A Sensory Earable Platform for Advanced Bio-Sensing and On-Device Machine Learning*. 2024. DOI: 10.48550/arXiv.2410.04775.

[11] Tobias Röddiger, Michael Küttner, Philipp Lepold, Tobias King, Dennis Moschina, Oliver Bagge, Joseph A. Paradiso, Christopher Clarke, and Michael Beigl. “OpenEarable 2.0: Open-Source Earphone Platform for Physiological Ear Sensing”. In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 9.1 (Mar. 2025). DOI: 10.1145/3712069.

[12] Andrea Ferlini, Alessandro Montanari, Andreas Grammenos, Robert Harle, and Cecilia Mascolo. “Enabling In-Ear Magnetic Sensing: Automatic and User Transparent Magnetometer Calibration”. In: *2021 IEEE International Conference on Pervasive Computing and Communications*. PerCom ’21. Kassel, Germany, 2021, pp. 1–8. DOI: 10.1109/PERCOM50583.2021.9439112.

[13] Chiara Gazzola, Valentina Zega, Fabrizio Cerini, Silvia Adorno, and Alberto Corigliano. “On the Design and Modeling of a Full-Range Piezoelectric MEMS Loudspeaker for In-Ear Applications”. In: *Journal of Microelectromechanical Systems* 32.6 (2023), pp. 626–637. DOI: 10.1109/JMEMS.2023.3312254.

[14] Jian Gong, Xinyu Zhang, Yuanjun Huang, Ju Ren, and Yaoxue Zhang. “Robust Inertial Motion Tracking through Deep Sensor Fusion across Smart Earbuds and Smartphone”. In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5.2 (June 2021). DOI: 10.1145/3463517.

[15] Grzegorz Zieliński, Beata Pająk-Zielińska, and Michał Ginszt. “A Meta-Analysis of the Global Prevalence of Temporomandibular Disorders”. In: *Journal of Clinical Medicine* 13 (Feb. 2024), p. 1365. DOI: 10.3390/jcm13051365.

[16] Denk Fysio. *Profile of Simone Gouw, Orofacial Physiotherapist*. Accessed: 25-08-2025. URL: https://www.denkfysio.nl/meedenkers-de-experts/simone-gouw.

[17] Max Jacob Frederik van Oort, Gabriel Enrique Sáenz, Selina Tirtajana, and Przemysław Pawełczak. “EarMag: In-Ear Magnetosensing for Jaw and Head Gesture-Based Human-Computer Interaction”. In: *Proceedings of the 6th International Workshop on Earable Computing (EarComp ’25), co-located with the 2025 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp Companion ’25)*. Espoo, Finland, 2025. DOI: 10.1145/3714394.3757254.

[18] Max Jacob Frederik van Oort, Gabriel Enrique Sáenz, Selina Tirtajana, and Przemysław Pawełczak. *EarMag: In-Ear Magnetosensing for Jaw and Head Gesture-Based Human-Computer Interaction*. 2025. URL: https://github.com/jawsaver/earmag-ubicomp2025.

[19] Euihyeok Lee, Chulhong Min, Jaeseung Lee, Jin Yu, and Seungwoo Kang. “GrooveMeter: Enabling Music Engagement-aware Apps by Detecting Reactions to Daily Music Listening via Earable Sensing”. In: *Proceedings of the 31st ACM International Conference on Multimedia*. MM ’23. Ottawa ON, Canada, 2023, pp. 7728–7736. DOI: 10.1145/3581783.3611968.

[20] Yetong Cao, Huijie Chen, Fan Li, and Yu Wang. “CanalScan: Tongue-Jaw Movement Recognition via Ear Canal Deformation Sensing”. In: *IEEE Conference on Computer Communications*. INFOCOM ’21. Vancouver, BC, Canada, 2021, pp. 1–10. DOI: 10.1109/INFOCOM42981.2021.9488852.

[21] Balz Maag, Zimu Zhou, Olga Saukh, and Lothar Thiele. “BARTON: Low Power Tongue Movement Sensing with In-Ear Barometers”. In: *IEEE 23rd International Conference on Parallel and Distributed Systems*. (ICPADS ’17). Shenzhen, China, 2017, pp. 9–16. DOI: 10.1109/ICPADS.2017.00013.

[22] Kazuhiro Taniguchi, Hisashi Kondo, Mami Kurosawa, and Atsushi Nishikawa. “Earable TEMPO: A Novel, Hands-Free Input Device that Uses the Movement of the Tongue Measured with a Wearable Ear Sensor”. In: *Sensors* 18.3 (2018). DOI: 10.3390/s18030733.

[23] Jay Prakash, Zhijian Yang, Yu-Lin Wei, Haitham Hassanieh, and Romit Roy Choudhury. “EarSense: earphones as a teeth activity sensor”. In: *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*. MobiCom ’20. London, United Kingdom, 2020. DOI: 10.1145/3372224.3419197.

[24] Wei Sun, Franklin Mingzhe Li, Benjamin Steeper, Songlin Xu, Feng Tian, and Cheng Zhang. “TeethTap: Recognizing Discrete Teeth Gestures Using Motion and Acoustic Sensing on an Earpiece”. In: *26th International Conference on Intelligent User Interfaces*. IUI ’21. Texas, USA (virtually hosted), Apr. 2021. DOI: 10.1145/3397481.3450645.

[25] Garvit Chugh, Suchetana Chakraborty, and Sandip Chakraborty. “Unlocking Eye Gestures with Earable Inertial Sensing for Accessible HCI”. In: *17th International Conference on COMmunication Systems and NETworks*. COMSNETS ’25. Bengaluru, India, 2025, pp. 828–832. DOI: 10.1109/COMSNETS63942.2025.10885707.

[26] Feiyu Han, Panlong Yang, Yuanhao Feng, Haohua Du, and Xiang-Yang Li. “Exploring Earable-Based Passive User Authentication via Interpretable In-Ear Breathing Biometrics”. In: *IEEE Transactions on Mobile Computing* 23.12 (2024), pp. 15238–15255. DOI: 10.1109/TMC.2024.3453412.

[27] Tanmay Srivastava, Prerna Khanna, Shijia Pan, Phuc Nguyen, and Shubham Jain. “Mutelt: Jaw Motion Based Unvoiced Command Recognition Using Earable”. In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6.3 (Sept. 2022). DOI: 10.1145/3550281.

[28] Prerna Khanna, Tanmay Srivastava, Shijia Pan, Shubham Jain, and Phuc Nguyen. “JawSense: Recognizing Unvoiced Sound using a Low-cost Ear-worn System”. In: *Proceedings of the 22nd International Workshop on Mobile Computing Systems and Applications*. HotMobile ’21. Virtual, United Kingdom, 2021, pp. 177–178. DOI: 10.1145/3446382.3450270.

[29] Himanshu Sahni, Abdelkareem Bedri, Gabriel Reyes, Pavleen Thukral, Zehua Guo, Thad Starner, and Maysam Ghovanloo. “The tongue and ear interface: a wearable system for silent speech recognition”. In: *Proceedings of the 2014 ACM International Symposium on Wearable Computers*. ISWC ’14. Seattle, Washington, 2014, pp. 47–54. DOI: 10.1145/2634317.2634322.

[30] Gabriel Enrique Sáenz, Selina Tirtajana, David Benjamin Jaroch, and Joost Plattel. *Jaw Movement Tracking System and Method*. Patent Application EP4440414A1, 29 November 2022. URL: https://patents.google.com/patent/EP4440414A1/.

[31] Erika Bondareva, Elín Rós Hauksdóttir, and Cecilia Mascolo. “Earables for Detection of Bruxism: a Feasibility Study”. In: *Adjunct Proceedings of the 2021 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2021 ACM International Symposium on Wearable Computers*. UbiComp/ISWC ’21 Adjunct. Virtual, USA, 2021, pp. 146–151. DOI: 10.1145/3460418.3479327.

[32] Murtadha Aldeer, David Waterworth, Zawar Hussain, Tahiya Chowdhury, Christian Brito, Quan Z. Sheng, Richard P. Martin, and Jorge Ortiz. “MedBuds: In-Ear Inertial Medication Taking Detection Using Smart Wireless Earbuds”. In: *2nd International Workshop on Cyber-Physical-Human System Design and Implementation*. CPHS ’22. Milan, Italy, 2022, pp. 19–23. DOI: 10.1109/CPHS56133.2022.9804515.

[33] Marília Barandas, Duarte Folgado, Letícia Fernandes, Sara Santos, Mariana Abreu, Patrícia Bota, Hui Liu, Tanja Schultz, and Hugo Gamboa. “TSFEL: Time Series Feature Extraction Library”. In: *SoftwareX* 11 (2020), p. 100456. DOI: https://doi.org/10.1016/j.softx.2020.100456.

A

Additional Figures

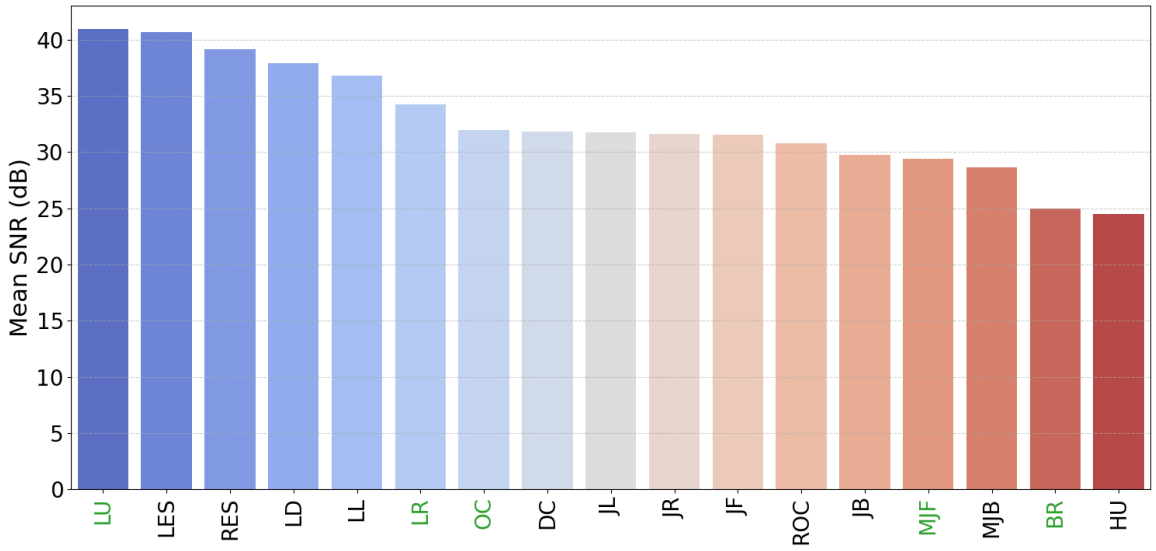


Figure A.1: Mean SNR per exercise for all user repetitions. The five exercises highlighted in green are the selected exercises.

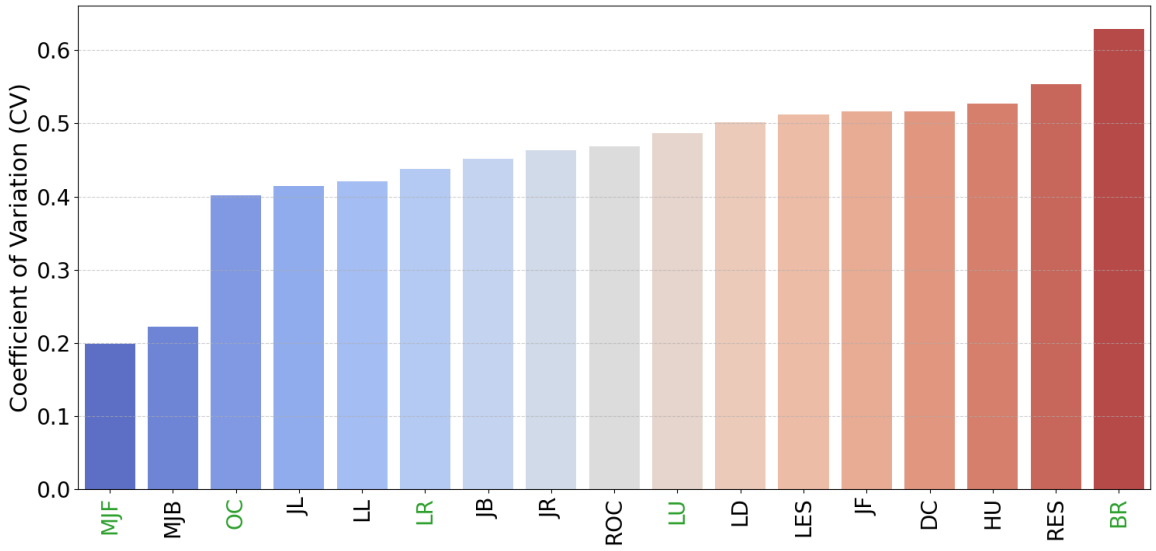


Figure A.2: Mean variability across repetitions of the same user per exercise (intra-user). The five exercises highlighted in green are the selected exercises.

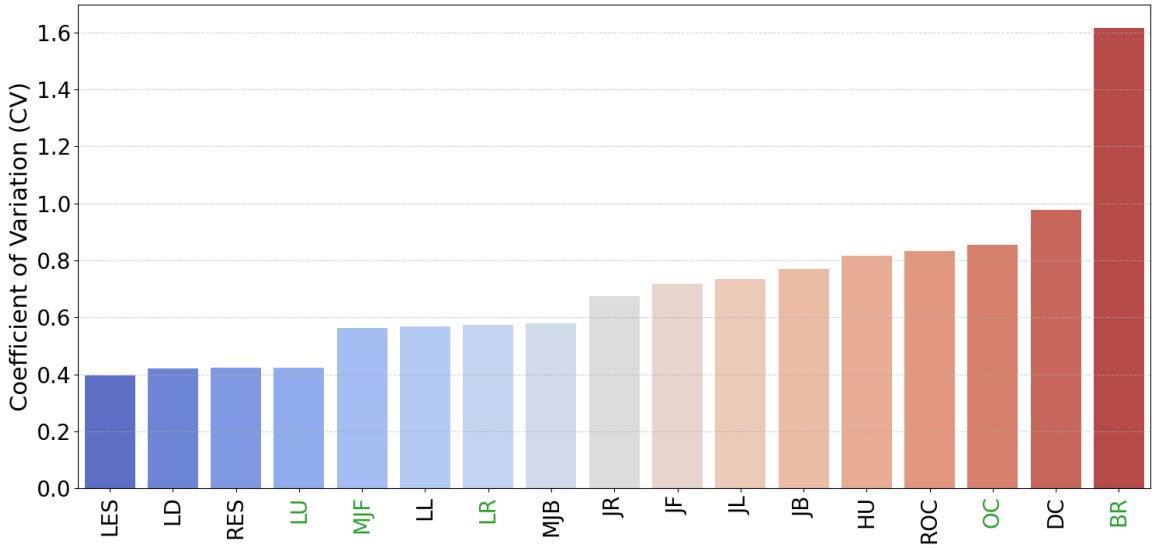


Figure A.3: Mean variability across users for all user repetitions per exercise (inter-user). The five exercises highlighted in green are the selected exercises.

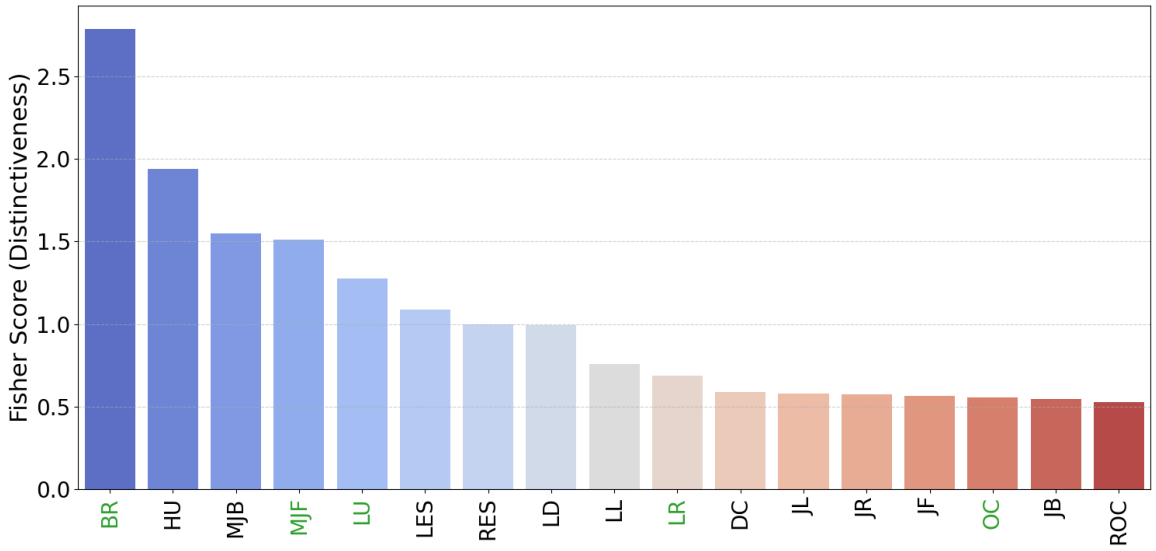


Figure A.4: Mean fisher scores for all user repetitions per exercise. The five exercises highlighted in green are the selected exercises.