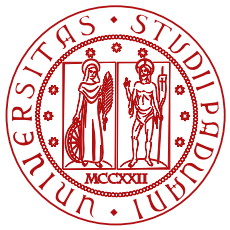


Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Master Thesis in ICT for Internet and Multimedia

Developing deep learning-based solutions for IMU time series with application in tele-rehabilitation

Master Candidate

Tommaso Urbani

Student ID 2045244

Supervisor

Prof. Giulia Cisotto

Co-supervisors

Dr. Eng. Giovanni Zanella

Dr. Eng. Alice Mantoan

Academic Year 2023/24

March, 7th 2024

Abstract

Tele-rehabilitation has emerged as a versatile and valuable approach, particularly in the post-Covid-19 era. Among the various techniques employed for data capture in tele-rehabilitation, Inertial Measurement Unit (IMU) sensors have gained prominence due to their ability to capture precise movements. These sensors are favored for their compact size, low power consumption and privacy-preserving features, making them a preferred choice in the field. Despite existing literature explores the segmentation of IMU time series in tele-rehabilitation studies, there is a noticeable gap in offering general solutions to address segmentation across a large variety of rehabilitation devices and motor exercises.

This thesis targets to address this gap by proposing a novel solution based on deep learning techniques to effectively segment IMU time series in the context of tele-rehabilitation. The proposed solution involves a series of pre-processing steps, a convolutional neural network (CNN), and post-processing procedures. The proposed model was then tested over a rich dataset obtained from the Henesis srl's ARC intellicare tele-rehabilitation device, incorporating data from 62 participants both healthy and pathological engaged in 41 distinct exercises, each monitored by three IMU sensors. The results indicate that the proposed model accurately identifies movements' initiation and completion with an average true positive rate of 90%; however, there is still room for improvement in mitigating a certain amount of false discovery rate (largely varying depending on the specific exercise). This work contributed towards the improvement of the company's system, while future investigations will involve a comparative analysis between the proposed model and the existing company solution. Moreover, it can also have an impact on other systems using IMU sensors to monitor rehabilitation treatments.

Sommario

La tele-riabilitazione si è affermata come un approccio versatile e prezioso, soprattutto nell'era post-Covid-19. Tra le varie tecniche impiegate per la raccolta dati nella tele-riabilitazione, i sensori di Unità di Misura Inerziale (IMU) hanno acquisito rilevanza grazie alla loro capacità di catturare movimenti precisi. Questi sensori sono preferiti per le loro dimensioni compatte, basso consumo energetico e funzionalità di tutela della privacy, rendendoli una scelta preferita nel settore. Nonostante la letteratura esistente esplori la segmentazione delle serie temporali di IMU negli studi sulla tele-riabilitazione, è evidente una lacuna nell'offrire soluzioni generali per affrontare la segmentazione su una vasta gamma di dispositivi riabilitativi e di esercizi motori.

Questa tesi si propone di colmare questa lacuna proponendo una soluzione innovativa basata su tecniche di apprendimento profondo per segmentare efficacemente le serie temporali di IMU nel contesto della tele-riabilitazione. La soluzione proposta coinvolge una serie di passaggi di pre-elaborazione, una rete neurale convoluzionale (CNN) e procedure di post-elaborazione. Il modello proposto è stato poi testato su un ricco set di dati ottenuto dal dispositivo di tele-riabilitazione intellicare ARC di Henesis srl, incorporando dati provenienti da 62 partecipanti, sia sani che patologici, impegnati in 41 esercizi distinti, ognuno monitorato da tre sensori IMU.

I risultati indicano che il modello proposto identifica con precisione l'inizio e il completamento dei movimenti con un tasso medio di true positive del 90%; tuttavia, c'è ancora spazio di miglioraramento per ridurre la quantità di false discovery rate (varia ampiamente a seconda dell'esercizio specifico).

Questo lavoro ha contribuito al miglioramento del sistema dell'azienda, mentre le future indagini coinvolgeranno un'analisi comparativa tra il modello proposto e la soluzione aziendale esistente. Inoltre, può avere un impatto su altri sistemi che utilizzano sensori IMU per monitorare trattamenti riabilitativi.

Contents

List of Figures

List of Tables

List of Acronyms

IMU Inertial Measurement Units

MEMS Micro-Electromechanical Systems

HAR Human Activity Recognition

pwPD people with Parkinson's Disease

COV19 post-COVID-19

CNN Convolutional Neural Network

ReLU Rectified Linear Unit

FC Fully Connected

MSE Mean Square Error

ARC ARC intellicare

NHSCT Northern Health and Social Care Trust

CERM Ethical Committee of the Marche Region

DWT Dynamic Time Warping

VR Virtual Reality

AR Augmented Reality

TPR True Positive Rate

FDR False Discovery Rate

1 Introduction

In recent years, tele-rehabilitation has emerged as a promising avenue in the field of healthcare, particularly gaining traction in the aftermath of the Covid-19 pandemic [1]. With its flexibility and potential to reach a wide range of patients, tele-rehabilitation offers an innovative approach to delivering rehabilitation services remotely. This shift towards tele-rehabilitation has been facilitated by advancements in technology, particularly in the realm of wearable devices such as Inertial Measurement Unit (IMU) sensors.

The integration of IMU sensors in tele-rehabilitation programs has enabled the capture of valuable data pertaining to patient movement and exercise performance in a more ecological way, even outside laboratories. These data hold immense potential for improving the quality and effectiveness of rehabilitation interventions. However, despite the growing body of literature on tele-rehabilitation [2] [3], there remains a gap in the development of solutions for segmenting IMU time series data in the context of rehabilitation exercises with a high number of different exercises employed.

This thesis work seeks to address this gap by presenting a deep learning approach to segmenting IMU time series data in tele-rehabilitation applications. The primary objective of this study is to develop a system capable of effectively identifying and quantifying training movements repetitions based on IMU data. To achieve this objective, the work leverages deep learning techniques, specifically Convolutional Neural Networks (CNNs), which have demonstrated significant promise in analyzing sequential data such as time series [4].

The dataset used in this study was made available by a collaboration with

Henesis s.r.l., which collected it using their product named *ARC intellicare*. Between 2018 and 2022, Henesis (formerly Health Division of Camlin Italy Srl) conducted two studies to validate the effectiveness and adaptability of ARC in diverse clinical settings. The first study, ARCANGEL, was a multicenter investigation involving post-stroke patients from Northern Health and Social Care Trust (NHSCT) of Northern Ireland and ASLTO3 of Pinerolo. The second, RICOMINCIARE, was a single-center study focused on individuals with Parkinsons disease and post-COVID-19 patients, supported by POR-FESR 2020, Università Politecnica delle Marche, and U.O. of Neurorehabilitation of Ancona. ARCANGEL, a prospective study, aimed to evaluate the feasibility of integrating ARC into clinical practice. The study focused on user and caregiver acceptance, satisfaction, and the impact of ARC on patients quality of life [5]. More than 10 million survivors require assistance with daily activities. Despite the proven benefits of formal rehabilitative therapies, less than a third of survivors receive these therapies, and accessibility varies based on geography and socioeconomic status. The study provided valuable insights into the real-world implementation of ARC, shedding light on its potential benefits in enhancing rehabilitation outcomes for post-stroke patients. In response to these challenges, the study introduces ARC, the goal is to extend rehabilitation access to a larger number of stroke patients, allowing for home-based conduction and monitoring. The second study: RICOMINCIARE was a single-center pre-/post-intervention pilot study aimed at verifying the feasibility and safety of the ARC system in the home rehabilitation of people with disabilities due to respiratory or neurological diseases. The study targeted people with Parkinsons Disease (pwPD) or post-COVID-19 (COV19) condition and an indication for exercise or home rehabilitation to optimize motor and respiratory function. The study [6] found that tele-rehabilitation through the ARC system is usable and safe for home rehabilitation in subjects with chronic motor and respiratory disabilities. Preliminary data suggest promising results on the effectiveness in subjects with post-COVID-19 condition or Parkinsons disease.

By leveraging this rich dataset, the proposed CNN-based model was trained and fine-tuned to obtain the best segmentation accuracy across the different types of exercises.

The structure of this thesis is organized as follows:

Chapter 2 - State of the Art: It provides an overview of the background and motivation behind the research, elucidating the significance of tele-rehabilitation and the role of IMU sensors in this domain. Also, offers a comprehensive review of existing literature, highlighting the current state-of-the-art in tele-rehabilitation and the methodologies employed for segmenting IMU time series data.

Chapter 3 - Methods: It describes the methodology adopted in this research, starting from the dataset description and the data acquisition process. Then the chapter explains the pre-processing steps, the model architecture, the training approaches and the evaluation metrics used to validate the proposed model.

Chapter 4 - Results: It presents the results obtained from the experiments conducted, offering insights into the performance of the proposed model across various exercises. The chapter also includes a discussion about the impact of the results.

Chapter 5 - Conclusions: It addresses the limitations of this work, proposing possible future research directions, and suggesting refinements to the proposed model.

2

State of the Art

Tele-rehabilitation, a subject of extensive research over the years [7] [8] [9], has gained considerable attention recently for its potential in delivering remote healthcare services. The impact of Covid-19 on the healthcare system prompted a sudden shift in rehabilitation practices, with physical therapists adopting tele-rehabilitation in innovative ways and with diverse patient populations [10].

Numerous studies in physical therapy emphasize the effectiveness of tele-rehabilitation, demonstrating its comparability with or superiority to in-person rehabilitation for conditions such as osteoarthritis, low-back pain, hip and knee replacement, multiple sclerosis, as well as in cardiac and pulmonary rehabilitation contexts [11].

Beyond mere accessibility [12] [13], tele-rehabilitation provides opportunities for continuous monitoring, personalized care, and timely interventions, ultimately contributing to improved patient outcomes. With ongoing technological advancements, the integration of emerging technologies like deep learning holds the promise of further enhancing the efficacy of tele-rehabilitation interventions.

A systematic review by Sarfo et al. [14] provides evidence to suggest that tele-rehabilitation interventions have either better or equal salutary effects on motor, higher cortical, and mood disorders compared with conventional face-to-face therapy.

2.1 IMU sensors

Telerehabilitation can be performed using webcams [15] [16], Virtual Reality Virtual Reality (VR) [17] [18], and motion sensors like Inertial Measurement Units Inertial Measurement Units (IMU) [19] [20] [21]. IMUs are devices that combine accelerometers, measuring linear acceleration, gyroscopes measuring angular velocity, and magnetometers measuring orientation with respect to the Earth's magnetic field. This combination of sensors allows IMUs to provide comprehensive information about an object's movement and orientation in three-dimensional space. Operating based on the principles of Micro-Electro-Mechanical Systems (MEMS), IMUs leverage the miniaturization of mechanical and electronic components. This miniaturization contributes to their versatility and applicability in various contexts, from healthcare to sports and beyond. Compact and lightweight, IMU sensors enable the capture of precise and dynamic movements, making them ideal for tracking human motion. Their widespread adoption is attributed to several advantages, characterized by their compact size and lightweight design, suitable for wearable devices and applications where space is a critical factor. Low power consumption enhances practicality, allowing for prolonged use without frequent battery replacements. Furthermore, IMU sensors excel in capturing both motion and orientation, offering a comprehensive perspective on physical activities. This versatility has led to their extensive utilization in various domains, including gait recognition [22], as well as the analysis of numerous other physical movements [23].

The advantages of using IMUs in tele-rehabilitation are multifaceted. Firstly, IMUs facilitate real-time and continuous monitoring of patients' movements, enabling healthcare professionals to assess rehabilitation progress remotely. This not only improves the efficiency of rehabilitation programs but also allows for timely adjustments to treatment plans based on objective data.

In the field of healthcare applications, IMUs have found widespread utility, addressing diverse needs in rehabilitation and beyond [24] [25]. The data generated by IMUs is subjected to various analyses, with segmentation emerging as a crucial aspect. Segmentation involves breaking down continuous time series movement data from IMUs into distinct segments or phases, aiding in the precise identification of specific movements or activities. The methodologies employed for segmentation range from conventional signal processing techniques [26] to

more contemporary machine learning and deep learning algorithms.

2.2 IMU SEGMENTATION

The usage of deep learning techniques in IMU data segmentation is gaining prominence in recent research [27] [28]. Deep Learning, particularly through the use of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), has showcased remarkable success in autonomously learning hierarchical features and capturing temporal dependencies within time series data. This adaptability makes deep learning a powerful tool for handling the intricate and dynamic nature of IMU data, leading to more robust and accurate segmentation outcomes.

A deep learning example application applied in IMU data segmentation is the research titled "Golf Swing Segmentation from a Single IMU Using Machine Learning" conducted by Kim Myeongsub and Park Sukyung [27] focused on accurately segmenting the golf swing into its five phases. Notably, each execution involved a single swing, simplifying the system's task of identifying distinct phases within the movement. Another investigation employing deep learning for IMU time series segmentation is work conducted by Luktuke Yadnyesh Y. and Hoover Adam, titled "Segmentation and Recognition of Eating Gestures from Wrist Motion using Deep Learning" [28]. This study aimed at detecting repetitive wrist movements over an extended period, specifically focusing on identifying instances when the subject was consuming a course. These projects highlight the versatility of IMU data segmentation across various contexts and durations. Another interesting project worth mentioning is ExerSense [29], which features a pipeline similar to the one proposed in this project. However, ExerSense utilizes a correlation-based approach instead of a machine learning approach. This distinction is possible because the ExerSense project involves a limited number of highly distinct movements for detection, providing an alternative perspective to segmentation methodologies.

While the previously discussed studies serve as great instances of IMU time series segmentation, it is noteworthy that within the domain of tele-rehabilitation, there exists a noticeable gap in research specifically targeting the detection and identification of a diverse array of rehabilitation exercises. One of the challenges

is to investigate the recognition of exercises performed by patients utilizing multiple IMUs, particularly when acquired without the direct physical assistance of a professional.

The overall goal of this thesis work is to perform a segmentation of data obtained from a set of IMUs within the domain of tele-rehabilitation. This involves a unique combination of IMU positioning data and the detection of a diverse set of movements not extensively covered in existing literature. The complexity of the targeted movements necessitates a tailored segmentation approach, making this research a valuable contribution to the evolving landscape of tele-rehabilitation interventions.

3

Methods

In the following sections, I present a comprehensive overview of the entire system, starting from the dataset description to repetition detection and counting. The foundation of this system lies in meticulous data analysis, where parameters are derived in a data-driven manner to tailor the approach to the inherent characteristics of each exercise data.

The chapter starts with the description of the data inside the dataset the acquisition system and the description of the library of exercises composing the dataset in order to understand the level of complexity and variety of the raw data.

The chapter continues with a detailed exploration of the pre-processing phase in Section 3.2 while Section 3.3 introduces the model architecture employed. Training approaches are elucidated in Section 3.4.

Section 3.5 is dedicated to the post-processing stage, a critical component in refining the model's predictions. In Section 3.6, the counting-based metrics are introduced to evaluate the accuracy of predicted repetitions.

3.1 ARC intellicare

The thesis was conducted in collaboration with Henesis s.r.l., a research company based in Parma, under the supervision of Giovanni Zanella and Alice Mantoan. The company provided access to the dataset of inertial signals acquired by their product, ARC intellicare (ARC).

ARC is certified as a Class I Medical Device according to MDR 2017/745 for use both in the clinic and at home, this device allows healthcare facilities and professionals to integrate tele-rehabilitation programs into standard in-person care, motivating patients with new and stimulating activities that can be performed autonomously and safely at home.

The device designed for patient use consists of hardware components and an integrated software, specifically:

- a set of 5 wearable sensors
- a tablet where a dedicated App runs
- a charging station
- a web portal

Allowing patients to follow their personalized rehabilitation plan without the need to travel toward rehabilitation centers while being constantly supervised remotely by professionals.

Thanks to the use of inertial sensors and validated artificial intelligence algorithms, ARC analyses movement in real-time. In addition, the device allows for the collection of patient-reported outcomes. Using validated clinical scales, the patient reports their feedback on their post-therapy health status and the perceived levels of pain and fatigue.

This section focuses into the core aspects of ARC, providing insights into the acquisition system, the acquisitions history, and a dataset description.

3.1.1 The Acquisition System

The ARC acquisition system is a sophisticated system designed to capture, process, and store motion data during rehabilitation exercises. Comprising a network of IMUs, the acquisition system meticulously records the dynamic movements of patients, offering a detailed representation of their motor activities.

ARC is equipped with wireless sensors that record accelerations and angular velocities related to the user's movement. These sensors are contained in special mounts that provide mechanical stability and are designed to be easily worn with one hand, ensuring they are not annoying when in contact with the skin as shown in Fig. 3.1 3.23.4 3.3.

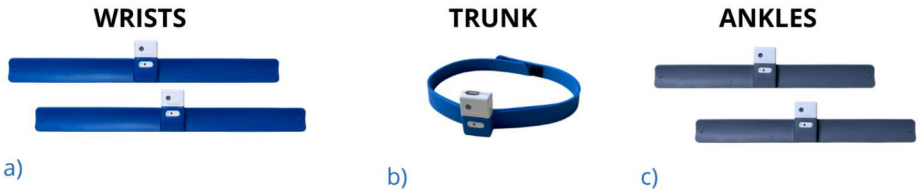


Figure 3.1: ARC Wireless Sensor Setup. a) Wrist supports with inertial sensors. b) Collar with inertial sensor. c) Ankle supports with inertial sensors.

Because ARC enables monitoring of both upper and lower limb movement and posture, the sensors are attached to wrist cuffs, ankle supports, and a collar. To facilitate easy wearing and removal, users can take advantage of their mechanical properties as demonstrated in Figs. 3.2, 3.3, and 3.4. The wrist cuff and ankle support, in fact, close on their own at the wrist and ankle, respectively, following a light blow. The outer covering is made of silicone (biocompatible), a material easily sanitized with a soft cloth soaked in an alcohol-based cleaning solution. The sensor support is designed to ensure it does not generate heat, enhancing comfort for patients.

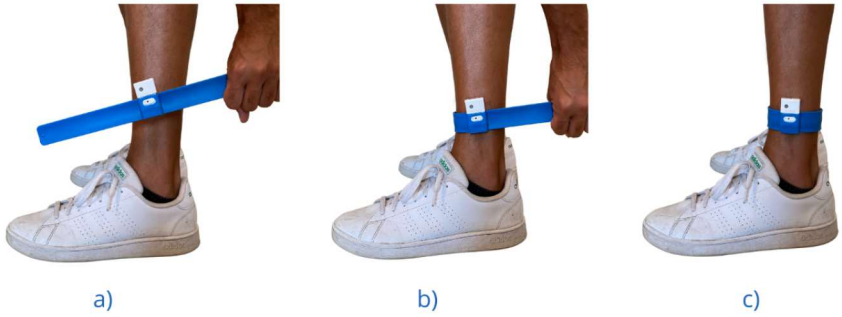


Figure 3.2: Ankle Support. a) Positioning of the ankle support. b) Activation of automatic closure and winding of the ankle support. c) Final position of the ankle sensor.

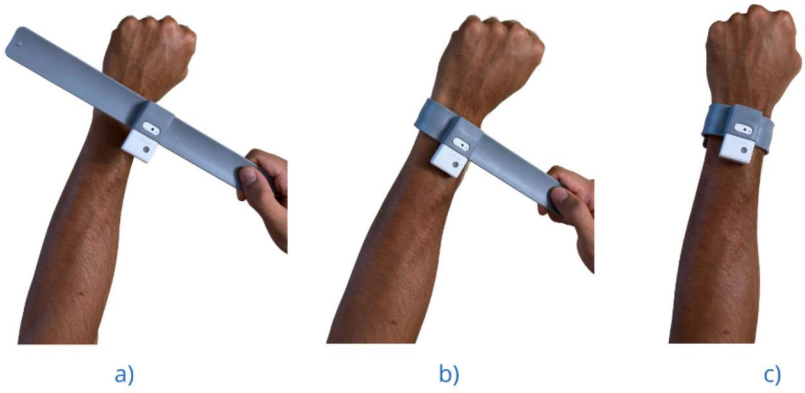


Figure 3.3: Wrist Support. a) Positioning of the wrist support. b) Activation of automatic closure and winding of the wrist support. c) Final position of the sensor at the wrist.

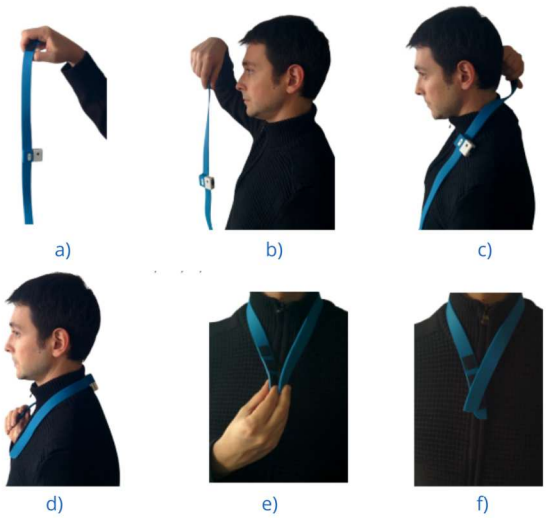


Figure 3.4: Trunk collar. Procedure for properly wearing the collar equipped with the inertial sensor.

The ARC acquisition system allows healthcare professionals to monitor and assess patients' exercises remotely. The system is non-intrusive, with patients wearing IMUs that house accelerometers and gyroscopes, providing a comprehensive view of both linear accelerations and angular velocities.

3.1.2 Dataset Description

The ARC dataset, is composed by data from multiple patients performing a set of exercises from a comprehensive collection.

Exercise Library and Patient Dataset

The exercise library, which encompasses over 60 activities, was meticulously crafted by the clinical team at the Neurorehabilitation Clinic, University Hospital Ospedali Riuniti di Ancona [30].

The patients' dataset utilized in this project originated from two studies: a clinical trial conducted in 2021, in collaboration with the Università Politecnica delle Marche. The clinical trial, entitled Home REhabilitation and Monitoring of People in post-covid Condition Through ARc-inTellicare Platform (RESTART/RICOMINCIARE) (RICOMINCIARE) (ClinicalTrials.gov Identifier: NCT05074771 [31]), gained approval from the Ethical Committee of the Marche Region (CERM). This study enrolled 21 out of 23 eligible patients, including 11 COV19 and 10 pwPD. Participants underwent training for ARC usage and received an ARC unit for independent use at home for 4 weeks, with 45-minute sessions, 5 days a week, tailored to respiratory and motor rehabilitation. The second study: ARCANGEL, detailed in Section 1, recruited 41 patients that have received an ARC device to use at home over a span of 6 months. This study addresses the significant challenges faced by stroke survivors, with approximately 36% experiencing significant disabilities five years post-stroke.

Moreover, it is essential to note that the entire exercise library has undergone clinical validation. The exercises featured in the library are part of a booklet [30] sponsored by SIMFER, the Italian Society of Physical Medicine and Rehabilitation. This sponsorship ensures that the exercises meet rigorous standards in the field of physical medicine and rehabilitation.

For the benefit of the public, the entire booklet, along with videos demonstrating each exercise, is made accessible through the platform created by the Polytechnic University of Marche and the Neurorehabilitation Clinic of the Ospedali Riuniti di Ancona. [1].

For the purpose of this project, emphasis is placed on exercises necessitating

[1] https://www.rehab-univpm.it/public/#/covid

the use of inertial sensors for execution. It is noteworthy that certain exercises, such as those focused on respiratory and stretching routines, do not require wearable devices, and consequently, their data are not included in the project dataset.

All exercises within ARC are presented through video tutorials where a therapist demonstrates correct execution, enabling patients to review and perform them accurately. The exercises are categorized into six groups: mobility, coordination, core stability, respiratory, static stretching, and strengthening. Targeted body areas include upper limbs, lower limbs, trunk, and the full body. Exercises are further differentiated based on specific patterns:

- **Bilateral Symmetric (BS)**: Symmetric movement of both arms or ankles.
- **Bilateral Alternated (BA)**: Alternating movement of two limbs.
- **Mono-lateral Left (ML)**: Involves only the left side of the body.
- **Mono-lateral Right (MR):** Targets the right side of the body.
- **Respiratory (RE)**: Involves respiratory movements.
- **Static (ST)**: Includes stretching exercises.

Some exercises may require additional equipment, such as a chair, a stick, an elastic band, weights, a bed, or a step. To simplify references, a system of abbreviations has been adopted in the dataset, the name of each exercise is divided in different parts: the first indicates the targeted body area, the second part indicates the pattern, and the last part stands for the exercise number. As example: exercise F.BA.001 means that the configuration is full body (F), the pattern is Bilateral Alternated (BA) and the code number is 001.

The following tables present the subset of 41 exercises selected for this project. Certain exercises were omitted from the selection as they do not necessitate the use of IMU sensors due to the nature of the movements. Each table includes the identification code of the exercise, an image illustrating the movement, and a description. This visual representation is essential for understanding the variety and complexity of the movements that the ARC system captures.

ID	IMAGE	DESCRIPTION
F.BA.001		Coordination while seated
F.BA.002		Supine coordination
F.BA.003		Coordination in orthostatism with one-support
F.BA.004		Lateral flexion-extension with stick in orthostatism
F.BA.005		Maintaining balance in single support with implement

Table 3.1: Exercise library - Full Body. Exercises F.BA.001-5 with respective image and description

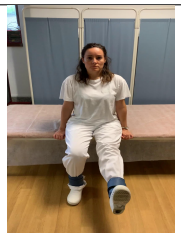
ID	IMAGE	DESCRIPTION
L.BA.001		Weight transfer on the lower limbs
L.BA.002		Lateral slides
L.BA.003		Rear slides
L.BA.004		Step forward
L.BA.005		Step back
L.BA.006		Side step
L.BA.007		Hip flexion
L.BA.008		Knee flexion-extension with resistance

Table 3.2: Exercise library - Lower Body. Exercises L.BA.001-08 with respective image and description

ID	IMAGE	DESCRIPTION
L.BA.009		Step
L.BA.010		Forward lunge without support
L.BA.011		Rear lunge without support
L.BA.012		Side lunge without support
L.BA.013		Single standing station with arms at 90°.
L.BA.014		Single standing station and support
L.BA.015		Tandem with support
L.BA.016		Tandem abducted limbs

Table 3.3: Exercise library - Lower Body. Exercises L.BA.009-16 with respective image and description

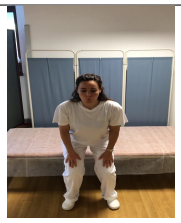
ID	IMAGE	DESCRIPTION
L.BA.017		Maintaining balance in single-support/hip flexion
L.BS.001		Squat
L.BS.002		Elevation on the forefoot
T.BA.001		Trunk rotations from standing combined with breathing exercises
T.BA.002		Seated trunk rotations combined with breathing exercises
U.BA.004		Seated arm extension
U.BA.005		Coordination in orthostatism with double support

Table 3.4: Exercise library - Trunk and Upper Body. Exercises L.BS, T.BA and U.BA with respective image and description

ID	IMAGE	DESCRIPTION
U.BA.006		Trunk rotation and flexion combined with breathing exercises
U.BS.002		Elevation of arms with stick from supine
U.BS.003		Elevation of arms sitting with stick
U.BS.004		Elevation of arms standing with stick
U.BS.005		Arm push exercise
U.BS.006		Seated arm abduction with elastic bands
U.BS.007		Extrarotation of shoulder with 90-degree elbow

Table 3.5: Exercise library - Upper Body. Exercises U.BA and U.BS with respective image and description

ID	IMAGE	DESCRIPTION
U.BS.008		Extrarotation of shoulder with arms abducted to 45 degrees
U.BS.009		Extrarotation of shoulder with arms abducted to 90 degrees
U.BS.010		Extrarotation of shoulder with elastic resistance
U.BS.011		Elevazione e abduzione braccia
U.ML.001		Extrarotation of left shoulder in decubitus with or without resistance
U.MR.001		Extrarotation of right shoulder in decubitus with or without resistance

Table 3.6: Exercise library - Upper Body. Exercises U.BS, U.ML and U.MR with respective image and description

The dataset is composed by repetitions performed by various individuals, including patients or healthy, each engaging in one exercise at a time. To provide a clearer overview of the database, (Figure 3.5) illustrates the distribution of execution for every exercise. This plot reveals substantial variations in the number of available samples for each exercise, raging from the highest with 283 executions to the lowest with 15.

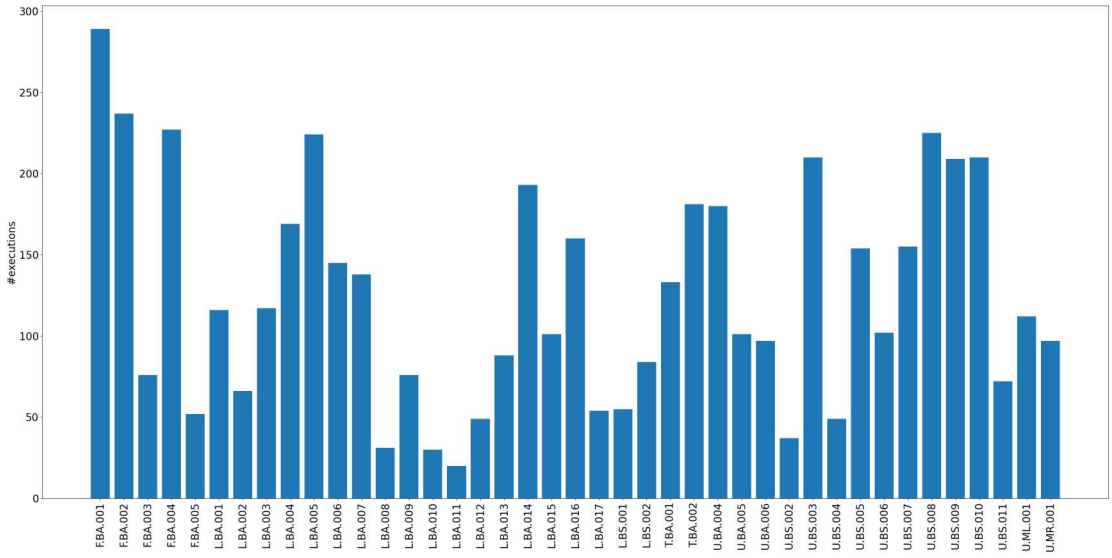


Figure 3.5: Number of Executions for every Exercise

As depicted in Figure 3.6, a crucial insight lies in the contrast between the number of executions performed by patients and those by healthy individuals for each exercise. A consistent range of 10 to 20 executions from healthy individuals for every exercise. However, the significant disparity arises in the proportion of these executions relative to the overall count, fluctuating from 7% to over 50% across exercises. Understanding the implications of this distribution is essential.

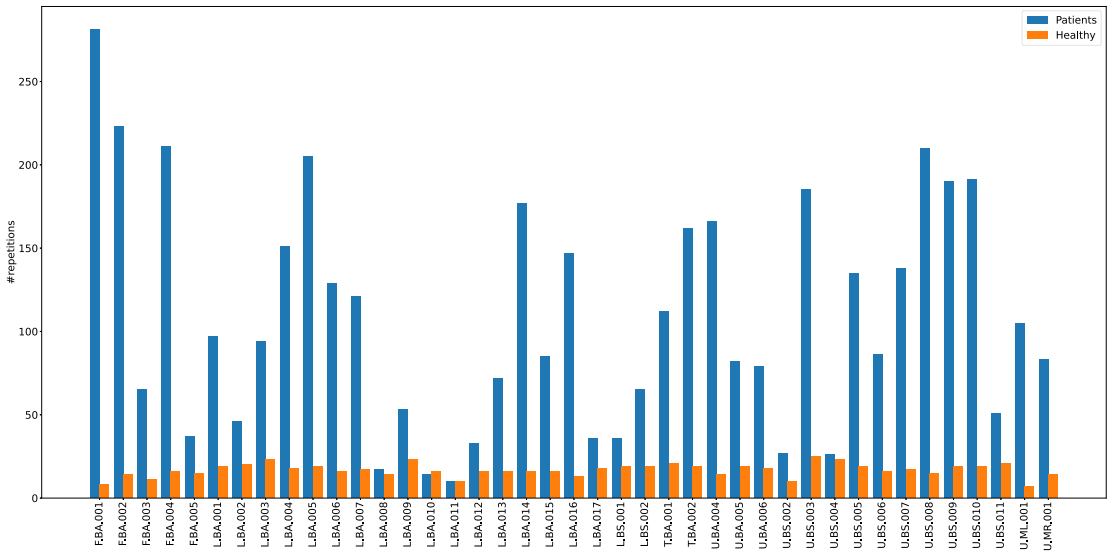


Figure 3.6: Comparison of Executions per Exercise Between Healthy Individuals and Patients. Healthy in blue and Non-Healthy in orange

Input Data Structure

To facilitate a clear understanding of the dataset and subsequent analysis, it is crucial to clarify some key terminology:

- **Exercise:** Refers to a specific movement or activity that a patient is required to perform.
- **Repetition:** Denotes the iteration of a particular exercise, capturing the completion of a full cycle of the movement.
- **Execution:** Represents the data recorded from the IMU sensors during the performance of an exercise. Each execution encompasses multiple repetitions, providing a comprehensive dataset for analysis.
- **Phase:** In the context of alternated exercises, where movements involve different sides of the body, a repetition is divided into two phases one for each side.

Each **execution** in the ARC dataset involves an exercises performed multiple times (**repetitions**) by a patient, providing a diverse and comprehensive set of motion data for analysis.

For each execution, data is collected from three IMU devices, each operating at a frequency of 50 Hz. Positions and orientation axes of the IMU sensors are illustrated in Figure3.7. The figure does not show a specific **configuration**, it illustrates the positions that the sensors can have. There are two different configurations: the Upper Body configuration, with a sensor placed in the trunk

and the remaining two sensors are positioned on the wrists; the Lower Body configuration, also a sensor is placed in the trunck but the two additional sensors are located on the ankles.

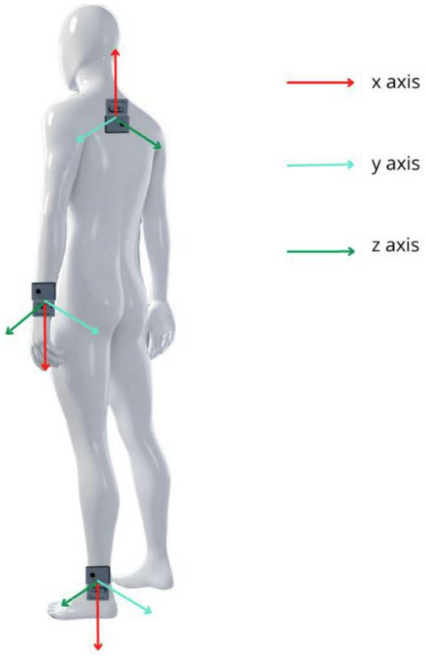


Figure 3.7: Sensor positions and orientation axes

For each timestamp there are three IMU devices recording data, each equipped with two sensors (accelerometer and gyroscope), and each sensor capturing data along three axes (x, y, and z), resulting in a total of 18 **channels** for each timestamp. The number of channels is given by the formula:

$$Num\_Channels = Num\_Devices \times Num\_Sensors \times Num\_Axes$$

This data structure allows for an exploration of patients' movements, enabling the development of machine learning algorithms that can discern intricate patterns and variations across different exercises and patients. To have a claer view of the data structure, refer to Figure 3.8.

DEVICE	SENSOR	AXIS	SIGNAL
Trunk	Accelerometer	X	
		Y	
		Z	
	Gyroscope	X	
		Y	
		Z	
Left	Accelerometer	X	
		Y	
		Z	
	Gyroscope	X	
		Y	
		Z	
Right	Accelerometer	X	
		Y	
		Z	
	Gyroscope	X	
		Y	
		Z	

Figure 3.8: Visual representation of the data structure in the ARC dataset.

To visualize the IMU data recorded for each execution, the company developed a tool called Annotator. Figure 3.9 illustrates the IMU signal for one accelerometer and one gyroscope along each axis (x, y, and z). The rectangles in the figure denote annotated repetitions, while the dashed lines indicate a phase change.

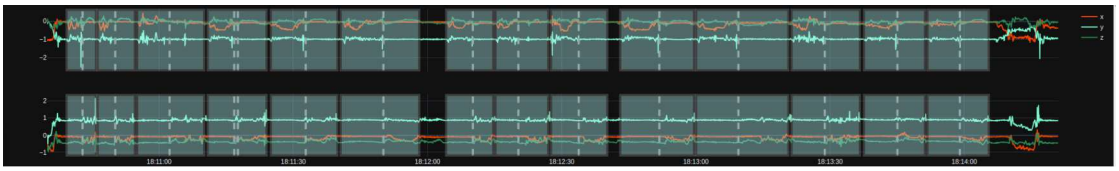


Figure 3.9: Visual representation of the IMU data from one accelerometer and one gyroscope

3.2 Pre - processing

The effectiveness of any machine learning model hinges significantly on the meticulous execution of pre-processing steps. These initial phases play a pivotal role in shaping the project's approach, based on the data analysis performed on the dataset.

The pre-processing pipeline initiates with a thorough examination of the existing data. The statistics extracted during this analysis serve a dual purpose: providing valuable insights into the nature of the problem at hand and serving as parameters within the system. Additionally, some of these statistics contribute to the standardization process applied to the data. Following standardization, the execution is segmented into windows, each serving as input for the model.

For each window, a corresponding ground truth is computed, serving as a label for loss computation and providing a visual representation of the model's output accuracy. This pre-processing methodology ensures that the model receives well-structured and standardized inputs, facilitating robust training and enhancing overall performance. By meticulously analyzing and standardizing data, our approach aims to optimize the model's ability to discern patterns, ultimately bolstering its predictive accuracy and generalization capabilities.

In Figure 3.10, the block diagram of the pre-processing pipeline is depicted, illustrating the key components: Extract Statistics, Data Standardization, Windowing, and Ground Truth.

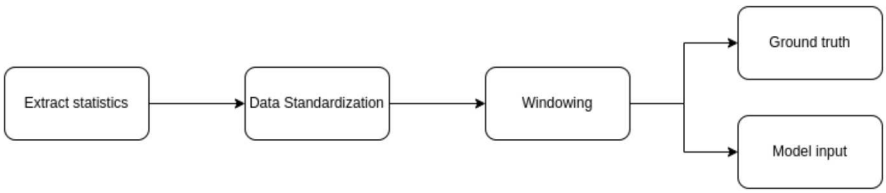


Figure 3.10: Pre-processing pipeline illustrated with a block diagram

3.2.1 Data Analysis

Data analysis and visualization were crucial steps, giving valuable insights for the subsequent modeling process. Various pieces of information were extracted, focusing on aspects that could enhance the model's performance. Statistics for the executions, divided by exercises, were obtained. For each of the 41 exercises, information on the execution duration, repetition duration, number of repetitions, and time distance between subsequent repetitions was collected. Additionally, visualizing the number of executions for each exercise provided essential context regarding the quantity of data available for each model, offering insights into their respective performances.

In the following figures, we present the statistical results instrumental in generating the parameter denoted as *mean_rep*, representing the mean duration of a repetition a parameter that influences pre-processing, model creation, and post-processing stages.

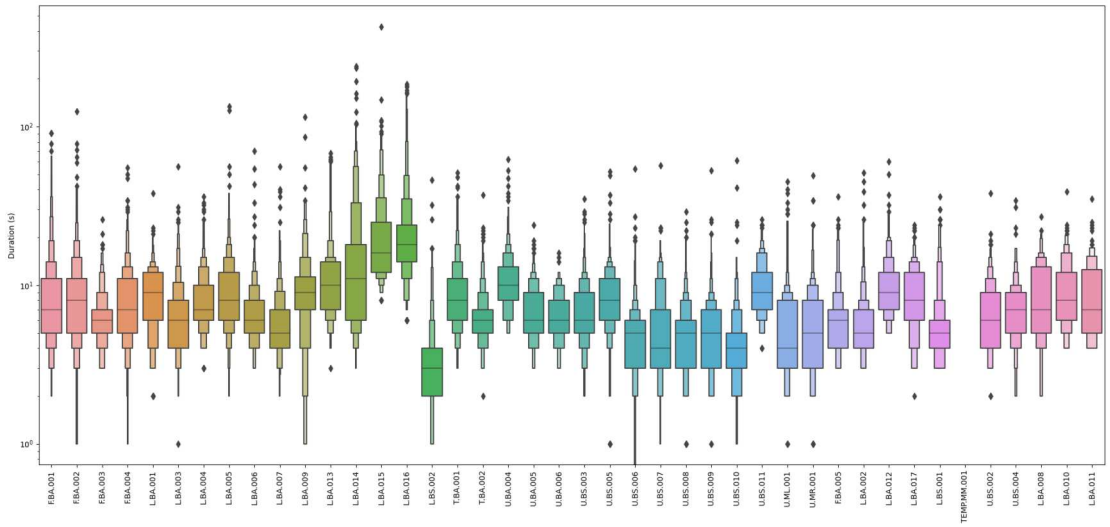


Figure 3.11: Statistical plot representing the repetition durations for each exercise

In Figure 3.11, the duration of each repetition, organized by exercise, is depicted. This statistical information is utilized to compute the parameter *mean_rep*.

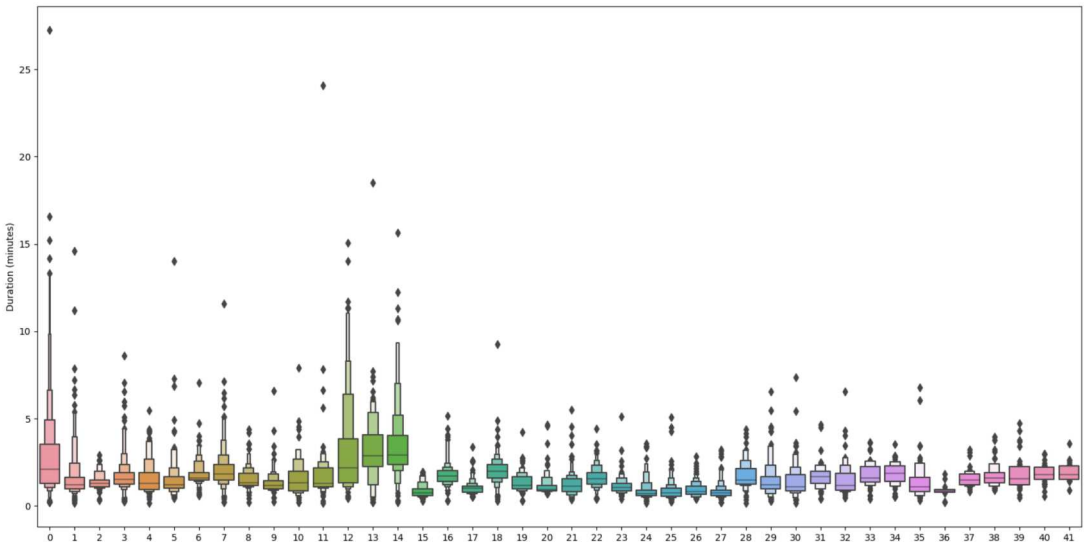


Figure 3.12: Statistical plot representing the execution durations for each exercise

Figure 3.12 presents details on the execution duration in seconds, categorized by exercise. This information is instrumental in determining the window parameter *w_len*. This parameter, calculated as the product of *mean_rep* and *mul*, is carefully chosen based on insights derived from the depicted statistics. Refer to Section 4.1.2 for a detailed exploration of this parameter selection process.

Data Standardization The initial data manipulation applied to the raw data involves the standardization process, aimed at enhancing the model's performance as documented in this paper [32]. This step begins with the normalization of the data for each of the 18 channels. Firstly, the mean (μ) and standard deviation (σ) of each channel are computed based on the previous data. Subsequently, the Z-score normalization is performed using the formula:

$$Z = \frac{X - \mu}{\sigma}$$

Here, X represents the raw data point, μ denotes the mean, and σ represents the standard deviation of the respective channel. The resulting Z-score provides a measure of how many standard deviations a data point is from the mean. This normalization process ensures that data across different channels are on a comparable scale, facilitating effective model training and convergence.

3.2.2 Sliding Window

Time series segmentation is a fundamental step in extracting meaningful patterns and dependencies within sequential data. In line with established practices in time series analysis [33], our approach employs the sliding window technique for segmentation. This widely adopted method involves systematically moving a fixed-size window through the time series data, with each window representing a distinct segment.

The key parameters governing this process are:

- `w_len` — Window Length
- `s` — Stride

The sliding window technique offers several advantages, as outlined in the literature [34]. It is effective in capturing temporal patterns, ensuring a smoother transition between adjacent segments, and facilitating the extraction of relevant features. The parameters `w_len` and `s` play a crucial role in defining the characteristics of the sliding window, with `w_len` determining the size of each window and `s` governing the step size during each iteration.

Our data-driven approach determines the choice of `w_len` based on the *mean_rep*, multiplied by *mul* . This selection aims to create windows large enough to capture relevant patterns while avoiding the exclusion of executions due to excessive padding, which could adversely affect model results.

The choice of the `s` parameter is made to incorporate overlapping windows. Overlapping windows, as suggested in [34], offer advantages such as capturing finer temporal details. However, to maintain a relatively slim model and avoid excessive redundancy in the data, the `s` parameter is chosen to provide the lowest number of overlapping windows necessary for the analysis.

The number of windows for each execution (N) is computed using the formula:

$$N = 1 + \frac{L - \texttt{w\_len}}{\texttt{s}}$$

where N is the number of windows, L is the total length of the execution, `w_len` is the length of each window, and `s` is the stride.

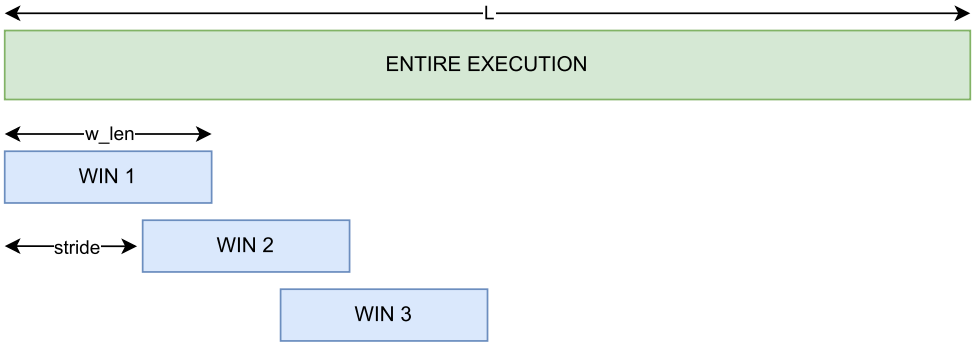


Figure 3.13: Representation of the windowing of an execution

3.2.3 Ground Truth

The ground truth in machine learning refers to the correct answer for a specific problem, its a critical aspect of the model development process, influencing both learning and evaluation. This choice is of paramount importance for two primary reasons.

Firstly, the ground truth serves as a benchmark for evaluating the model's performance. It provides a reference against which the predictions of the model can be compared, allowing for a systematic assessment of accuracy, precision, and recall. An accurate and representative ground truth is essential for meaningful comparisons between different models or variations of the same model. It ensures that the evaluation metrics used to guide the model's effectiveness reflect its ability to capture relevant patterns and make accurate predictions.

Secondly, the choice of ground truth plays a pivotal role in guiding the model's learning process. During training, the model learns from the provided ground truth to identify and understand patterns in the input data. The ground truth serves as a teacher, guiding the model toward the correct representation of phases, transitions, and other critical aspects of the data. An appropriately chosen ground truth enhances the model's ability to generalize well to unseen data, as it learns to recognize the underlying structure of the time series.

In this study, the term ground truth apply to data manually labeled by physicians for all the executions employed as the dataset. The physicians annotations consists into different types of markers:

- **Start Marker:** Indicates the beginning of a correctly performed repetition.
- **Phase Marker:** Pertains to the transition between phases within a repetition. This is specifically applicable to exercises with a Bilateral Alternated

pattern, as detailed in Section 3.1.3, where such exercises comprise two distinct phases.

- **End Marker:** Corresponds to the conclusion of the entire repetition.

Given these considerations, three distinct types of ground truths were introduced:

- **Rectangular Ground Truth:** This ground truth is represented by a rectangular signal that transitions to 1 when the repetition starts and returns to 0 when the repetition ends. However, it does not provide information about phase-changing points.
- **Ramp Ground Truth:** The ramp ground truth is characterized by a gradual increase from 0 to 1, reaching 1 when there is phase marker or the end marker. One drawback is the potential for not excessive distinction between phases and repetitions, complicating the learning process for the model.
- **Triangular Ground Truth:** In this ground truth, a triangular signal starts when the repetition begins, reaches its peak at the phase marker, and then gradually decreases to zero when the repetition ends. If the exercise doesn't include a phase marker, the peak will correspond to themiddle point between the start and the end of the repetition.

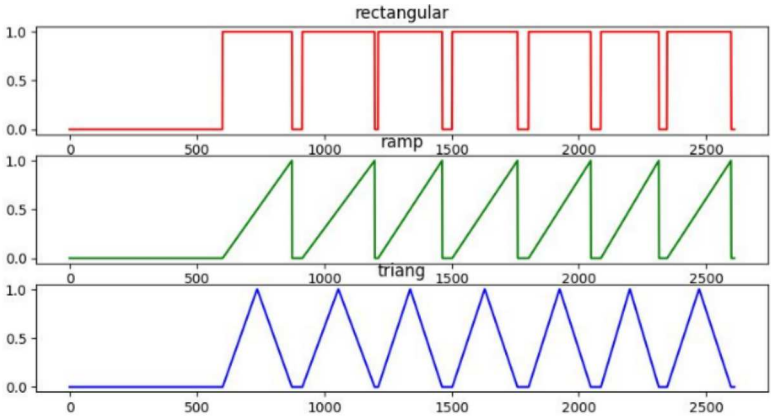


Figure 3.14: Displaying the different Ground Truths for the exercise U.BS.010 from the patient-ricominciare-U003

In the context of the proposed ground truths, the selection involves a trade-off between complexity and information. The ground truth must capture the nuances of repetitions and phase changes in a way that facilitates the model's understanding without introducing unnecessary complications. The chosen ground truth should strike a balance, enabling the model to learn the relevant features for both regression tasks and phase detection.

The triangular ground truth emerged as the most effective choice. Its simplicity ensures straightforward model learning, while its performance in regression and phase detection tasks indicates a clear representation of the underlying patterns. Thus, the ground truth becomes not just an evaluation metric but an integral part of the model's training, significantly impacting its overall effectiveness in understanding and predicting complex temporal relationships in the time series data.

3.3 Model

Deep learning has reshaped diverse domains by empowering computers to decipher intricate patterns and representations from data. Central to the paradigm of deep learning is the neural network, drawing inspiration from the intricate structure and functioning of the human brain. A pivotal architectural element within the realm of deep learning is the Convolutional Neural Network (CNN), meticulously crafted to efficiently process grid-like data, such as images and time series [35][36].

CNNs have showcased remarkable proficiency in capturing hierarchical features within data, rendering them a favored choice for tasks like image classification, object detection, and sequence analysis. The architecture comprises layers that progressively acquire and distill intricate patterns, enabling the model to discern hierarchical representations embedded in the input data.

The network architecture development used an approach starting from the simplest model and progressively increasing the complexity following the [37] approach. The architectures used in this system encompasses a convolutional layer succeeded by two fully connected layers. During the architecture development various CNN architectures were taken into consideration but overall this particular configuration yields optimal results, considering the limited dataset where a more complex architecture would introduce redundancy.

Figure 3.15 illustrates the complete model architecture. Starting from the input, the convolutional filters progressively extract features. The data is then flattened, and feed into two fully connected layers further process these features, culminating in the resulting output. This architectural overview provides a visual representation of the flow of information through the model.

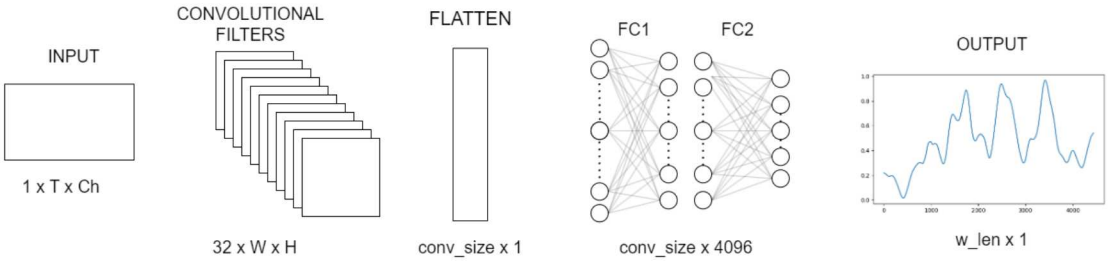


Figure 3.15: Abstract representation of the entire model architecture

The subsequent sections delve into the specifics of the network architecture.

Section 4.2.1 provides an in-depth exploration of the Convolutional Layer, unraveling its role in extracting essential features. Following that, Section 4.2.2 dissects the intricacies of the Fully Connected Layer, shedding light on how each component contributes to the overall effectiveness of the model.

3.3.1 Convolutional Layer

Deep learning, particularly CNN, has demonstrated remarkable success in various applications, leveraging their ability to automatically learn hierarchical representations from data [38]. The convolutional layer is a pivotal component of CNNs, responsible for capturing local patterns and features within the input data. In this section, we delve into the specifics of the convolutional layer design, emphasizing the data-driven choices made for kernel size, padding, and stride.

The convolutional layer operates by convolving filters or kernels over the input data to extract features. The kernel size is a critical parameter, influencing the receptive field and the network's ability to capture intricate patterns. Importantly, our approach employs a data-driven strategy for determining the kernel size. Specifically, the size is set as $(\text{mean\_rep} \times 3, 18)$, where mean_rep is the mean duration of the repetitions for that specific exercise, and 18 is the number of channels in the input data.

The advantages of using a kernel size varying from exercises to exercise are multiple. First and foremost, it allows the model to adapt to the temporal characteristics of the data. By using the mean duration of repetitions for each exercise, the kernel becomes responsive to the inherent structure of the time series. This kernel size adaptability between different exercises enables the model to learn multiple features simultaneously, potentially capturing complex temporal relationships.

Moreover, a kernel size of $(\text{mean\_rep} \times 3, 18)$ facilitates the learning of correlations between adjacent repetitions since it should detect multiple repetitions with the same kernel. This is particularly valuable in the context of exercise data, where understanding the temporal dynamics and transitions between repetitions is crucial. By aligning the kernel size with the mean duration of repetitions, the model gains the capability to recognize and leverage these temporal correlations effectively.

The channel dimension of the output (C_{out}) is the number of filters applied, which is typically determined by the design choices and requirements of the model; in this case, 32 was chosen. This choice permits the model to learn 32 different features simultaneously with the same temporal resolution, considering the system deals with 18 channels, and each exercise has different characteristics, where repetitions can be observed from different channels or multiple channels simultaneously.

This approach to the convolutional layer design not only optimizes the model for the specific temporal characteristics of the exercise data but also enhances its capacity to learn intricate features and correlations within the time series.

3.3.2 Fully Connected Layer

After the convolutional layer, the extracted features are flattened to serve as input for fully connected layers, allowing the model to capture higher-level representations and make predictions based on the learned features [39]. The system architecture incorporates two fully connected layers, followed by Rectified Linear Unit (ReLU) activation and dropout regularization.

This design choice is influenced by research findings that explore the relationship between Convolutional Neural Networks (CNNs) and Fully Connected (FC) layers, as presented in the work by Basha et al. [40]. The authors investigate the impact of deeper architectures on the performance of CNNs with respect to FC layers, as well as how the depth and width of datasets influence CNN performance in the context of Fully Connected (FC) layers. This research contributes to the understanding of CNN architecture development, especially in the context of image classification.

Flattening The output from the convolutional layer, initially a three-dimensional tensor, undergoes flattening into a one-dimensional vector before entering the fully connected layers. This transformation is essential for converting the spatial and temporal features acquired by the convolutional layer into a format suitable for processing by densely connected layers.

First Fully Connected Layer (FC1) The flattened output from the convolutional layer is then fed into the first fully connected layer. The input size of this layer is determined by the result of the convolutional layer. The hidden size is set to 4096 neurons which ensures that this layer has at least two neurons for each input data point. This layer is strategically designed to capture complex relationships and patterns within the high-dimensional feature space derived from the convolutional layer. The ReLU activation function is applied element-wise to introduce non-linearity, enabling the model to learn more intricate representations.

Dropout Regularization To mitigate overfitting and enhance generalization capabilities, dropout regularization with a dropout rate (p) of 0.2 was applied following the first fully connected layer. Dropout randomly sets a fraction of input units to zero during training, reducing reliance on specific neurons and promoting a more robust learning process. The parameter p represents the probability of dropping out a neuron, and in this context, it was set to 0.2.

This approach is inspired by the work on dropout regularization [41], where dropout is used as a regularization technique. Dropout and other feature noising schemes control overfitting by artificially corrupting the training data [42]. For generalized linear models, dropout performs a form of adaptive regularization.

Second Fully Connected Layer (FC2) This layer has an input size of 4096 neurons and an output size of `w_len`, where `w_len` represents the chosen window length. The second fully connected layer further refines the learned representations and prepares the model for predicting the temporal characteristics of the time series data. The ReLU activation function is once again applied to introduce non-linearity.

The choice of `w_len` as the output size is significant as it aligns with the objective of creating predicted ground truth sequences. These sequences can then be compared with the actual ground truth, allowing the computation of the mean square error loss during the training process. This comparison forms a crucial step in training the model to accurately predict the temporal characteristics of the input time series data.

These fully connected layers, combined with appropriate activation functions and regularization techniques, empower the model to learn hierarchical representations and make accurate predictions based on the input time series data.

3.4 Training

This section provides a detailed description of the parameters used in the model training, their significance, and the optimal combinations identified through testing.

Train, validation and test split The executions for each exercise was divided into three sets:

- 70% allocated for the training set
- 20% designated for the validation set
- 10% reserved for the test set

The decision to separate the executions and not the windows in its entirety was made to ensure a fair test environment because this approach guarantees that, during training, the model does not learn from any windows belonging to executions on which the test was made. This careful partitioning also facilitates the prediction of entire executions, as all windows are inside within the same set.

Loss function The Mean Square Error (MSE) was chosen as the loss function for training. The MSE serves as a metric for evaluating the model's accuracy by measuring the average squared difference between predicted values and ground truth. This mathematical representation is defined as:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

where N is the number of samples, y_i is the ground truth, and $\hat{y}_i$ is the predicted value. The selection of this loss function is grounded in the nature of the data. Given that both the ground truth and the model output consist of arrays of floats, the MSE is well-suited for this scenario. Moreover, the wide useage use of MSE in time series analysis, as evidenced by references [43], [44], and [45], further justifies its adoption but possible alternatives could include the usage of Dynamic Time Warping (DWT) as shown in the study [46].

Epochs For the model training, a choice was made not to rigidly set a fixed number of training epochs. Instead, early stopping was employed as a regularization technique to prevent overfitting on the training set. In this strategy, the validation loss served as parameter for the early stopping mechanism. If the loss does not decrease consistently over several consecutive epochs, the training process is stopped. The term *patience* refers to the number of epochs to monitor, and for this project, given that at least 300 epochs were required for the model to achieve satisfactory results, a patience of value of 50 epochs was selected. This approach ensures a balance between model training thoroughness and the prevention of overfitting.

Learning rate The initial learning rate was dynamically determined for each model, tailored to the characteristics of the data employed in each exercise. This was achieved using the PyTorch Lightning function known as *LR_find*. Drawing inspiration from the methodology presented in the paper by Smith [47], this function executes a brief run, incrementing the learning rate after each processed batch while logging the corresponding loss. This dynamic process allows for the estimation of an optimal initial learning rate, adapting it to the specific nuances of the dataset under consideration.

Optimizer The selected optimizer for the model was the Adam Optimizer, a stochastic gradient descent method founded on adaptive estimation of first-order and second-order moments [48]. Renowned for its computational efficiency and minimal memory requirements, the Adam Optimizer stands out as the most widely employed optimizer in various applications.

The summarized parameters, outlined earlier, are presented in Tables 3.7, 3.8 and 3.9 :

Parameter	Value
Batch size	8
Loss function	MSE
Patience	50 epochs
Optimizer	Adam
Learning rate	10^{-6} to 10^{-3}
Train split	70%
Validation split	20%
Test split	10%

Table 3.7: Fixed parameters for Training

Parameter	Value
Conv kernel size	$(mean\_rep \times 4, 18)$
Conv stride	$(1, 1)$
Conv padding	$(0, 0)$
# conv filters	32

Table 3.8: Parameters for the convolutional layer

Parameter	Value
FC1 size	$(conv\_size, 4096)$
FC2 size	$(4096, w\_len)$
Dropout	(0.2)
Model output	$(w\_len, 1)$

Table 3.9: Parameters for the fully connected layers

Following this, the upcoming section will elaborate on the two distinct training approaches employed in the subsequent phases.

3.4.1 Baseline approach

Initially, the strategy involved training a separate model for each exercise. To implement this, we categorized all executions based on the specific exercise performed. Subsequently, the data for each exercise was partitioned into training, validation, and test sets, utilizing the previously defined parameters.

The model training process incorporated the *mean_rep* parameter calculated from all executions of the corresponding exercise, using both the training and validation sets. In Figure 3.16, we present the values of *mean_rep* for each exercise.

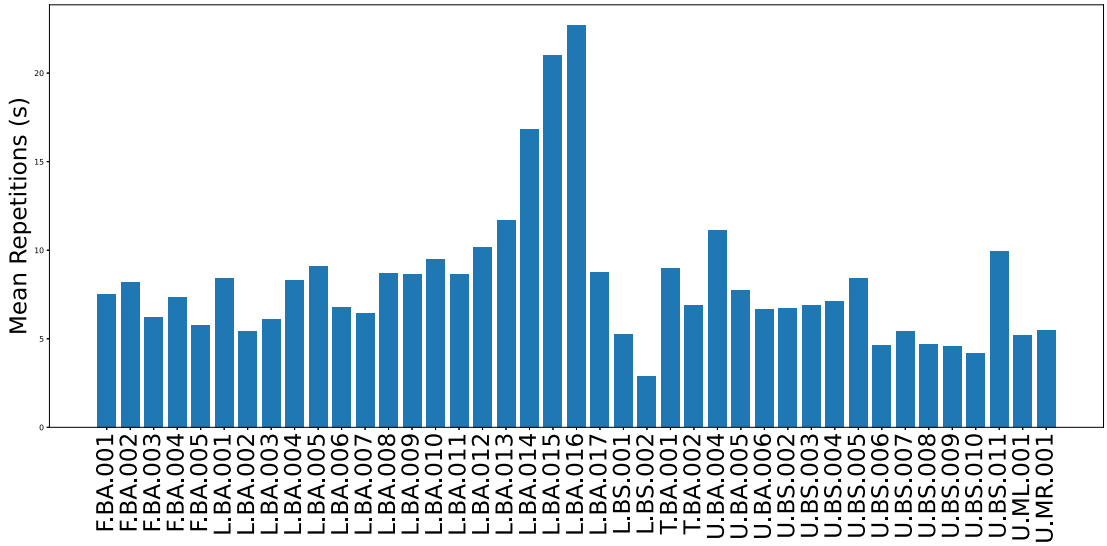


Figure 3.16: Parameter mean_rep for every exercise

This methodology ensured the creation of a dedicated model for each exercise, emphasizing specialization in learning exercise-specific features. However, a notable drawback emerged, as the executions for certain exercises lacked sufficient data for robust feature learning, as illustrated in Figure 3.5. To address this challenge, an alternative approach was adopted.

3.4.2 Grouping and Fine Tuning

The second approach used started from grouping the exercises into groups with similar *mean_rep*, then train a model for each group and finally starting from this model fine tune the model for each exercise.

Grouping To aggregate the exercises, various possibilities were considered. However, in alignment with the system approach to the data-driven parameter *mean_rep*, the decision was made to utilize it as the primary grouping parameter. This choice was taken since the system has the parameter *mean_rep* regulating the sizes of w_len, stride and the model architecture for each exercise. The distribution of *mean_rep* values for each exercise is visualized in Figure 3.16. Once *mean_rep* was established as the grouping parameter, the next step was to determine the grouping method.

Hierarchical clustering, as described by Nielsen [49], was employed based on the *mean_rep* parameter. This approach organizes exercises into a tree-like structure, grouping them according to their similarities, where similarity is defined by the values of *mean_rep*. In Figure 3.17, the color-coded *mean_rep* values for each exercise depict the resulting groups achieved through hierarchical clustering.

Throughout the development of this approach, various numbers of groups were considered. The hierarchical clustering algorithm, guided by the threshold set for the number of groups, underwent several tests. The optimal results were obtained when the system was configured to generate four distinct groups.

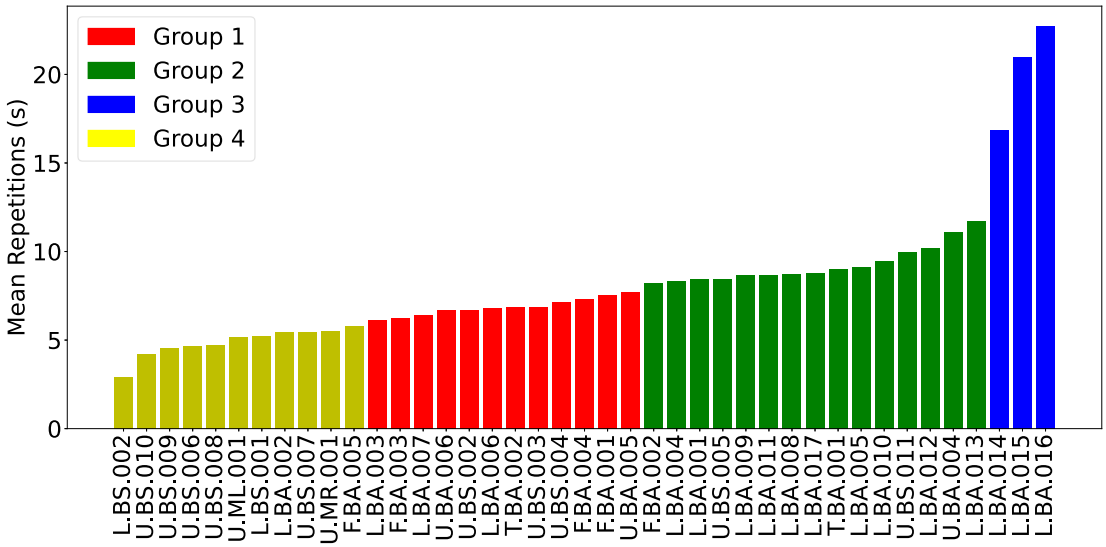


Figure 3.17: Color-coded representation of the *mean_rep* parameter for each exercise group.

Training The training process involved a procedure for each exercise within every group. A split was performed to create distinct training, validation, and

test sets for each exercise. Subsequently, the individual training sets for each exercise within a group were merged to form the comprehensive training set for that specific group. A similar consolidation process was applied to construct the validation set.

To tailor the input windows, determine the stride, and configure model layers, a *mean_rep* parameter for each group was extracted. The resulting hyperparameters for window length (**w_len**), stride, and kernel size for each group are detailed in Table 3.10.

	mean_rep	**w_len**	**stride**	**Kernel size**
Group1	243	972	486	(702, 18)
Group2	343	1372	686	(1029, 18)
Group3	462	1848	924	(1386, 18)
Group4	1009	4036	2018	(3027, 18)

Table 3.10: Hyperparameters for shaping input windows and configuring model layers in each group.

Upon completion of the training process, four distinct models, each corresponding to one of the designated groups, are generated. Figure 3.18 provides a systematic overview, illustrating the initial training progress.

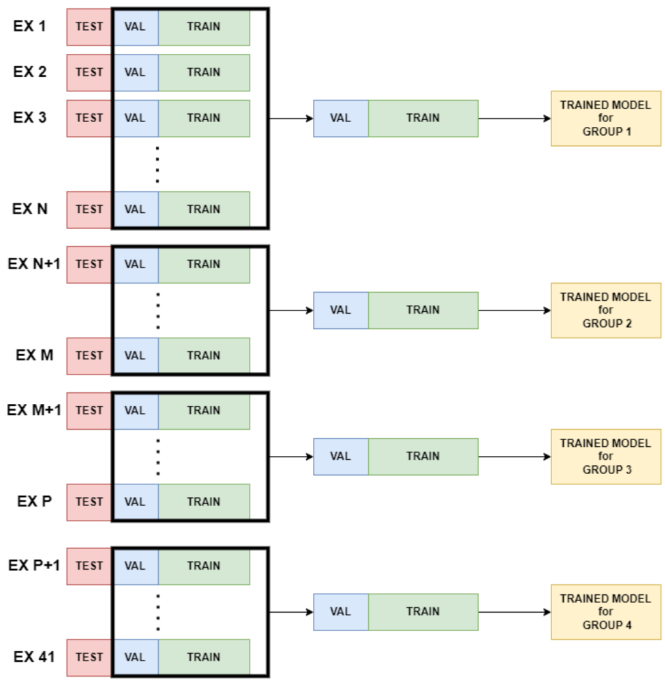


Figure 3.18: Group training

Fine Tuning Following the initial training, there exist four models, one for each group. Subsequently, each model undergoes fine-tuning with respect to every exercise within its designated group. The fine-tuning process utilizes both the training and validation sets of each exercise to adapt the model's weights.

Fine-tuning involves adjusting the pre-trained model's weights (the group model in the project context) to better suit a specific problem, in this case, the unique characteristics of each exercise. Figure 3.19 presents the completion of the scheme initiated in Figure 3.18, encompassing the fine-tuning phase.

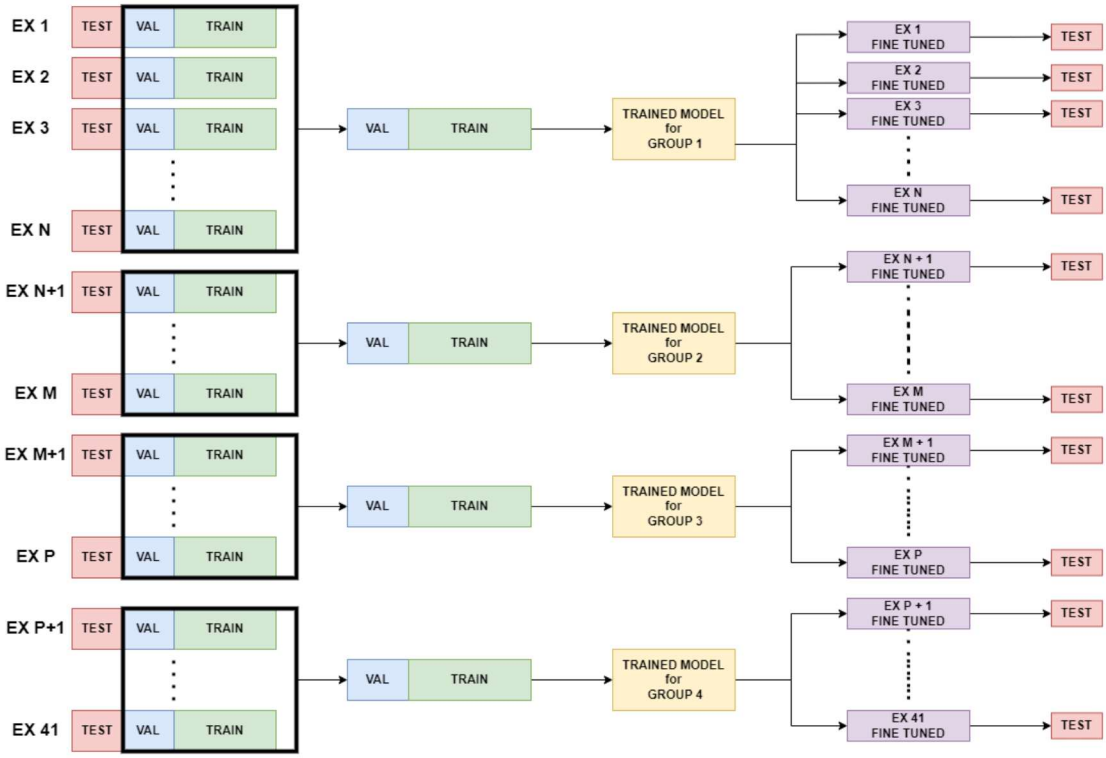


Figure 3.19: Grouping and Fine Tuning Overview

The subsequent section focuses into the post-processing steps, elucidating how the model's predictions are interpreted and refined to derive meaningful insights. The model's effectiveness and generalization capabilities are examined through evaluation metrics explained in Section 4.1.

3.5 Post-processing

The output of the CNN model has the same size of the ground truth described in Section 3.2.3, which is $(w\_len, 1$. The system involves a set post-processing steps, where the predictions on the test set are refined to extract meaningful insights and annotations. The process starts with the application of the trained model to the executions within the test set, marking the initiation of the post-processing pipeline. The model predictions serve as a foundation for a series of steps designed to enhance precision and interpretability.

The first step in post-processing is the application of a smoothing procedure to the model's output. This step offers several advantages, including the reduction of noise and the accentuation of underlying patterns.

Following smoothing, the post-processing pipeline incorporates peak detection algorithms, to identify key markers such as the start and end of repetitions and phase transitions. These markers serve as crucial reference points for further analysis and annotation.

The identified repetitions are computed using a custom filtering function. This function is designed to force the recognition of a specific pattern, i.e., start-phase-end as introduced in Section 3.2.3. The careful curation of detected points ensures that only complete repetitions contribute to the final analysis, enhancing the reliability and accuracy of the results.

The post-processing journey finalizes with the utilization of the predicted and filtered output to quantify repetitions and precisely locate them within the time series. This annotated information lays the groundwork for a comprehensive evaluation of the model's performance, providing valuable insights into its ability to discern and predict complex temporal relationships within the exercise data.

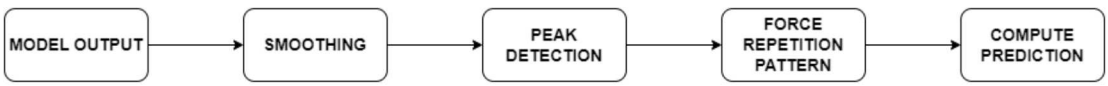


Figure 3.20: Post-processing pipeline

Smoothing

The initial post-processing step involves smoothing the output to impose a degree of continuity onto the predictions. This smoothing operation enhances the coherence of the temporal dynamics representation, establishing a foundation for subsequent analyses.

To achieve this, a moving average with a temporal size of 1 second (equivalent to 50 timestamps) is applied. This temporal smoothing not only refines the predictions but also facilitates the exploration of the underlying patterns and trends within the data.

Peak Detection

After applying the smoothing operation to the model's output, as illustrated in Figure 3.21, the temporal dynamics become more discernible, revealing distinct points corresponding to the start of repetitions, phase transitions, and the conclusion of executions. The enhanced clarity of these features sets the stage for automated peak detection, a crucial step in refining the predictions.

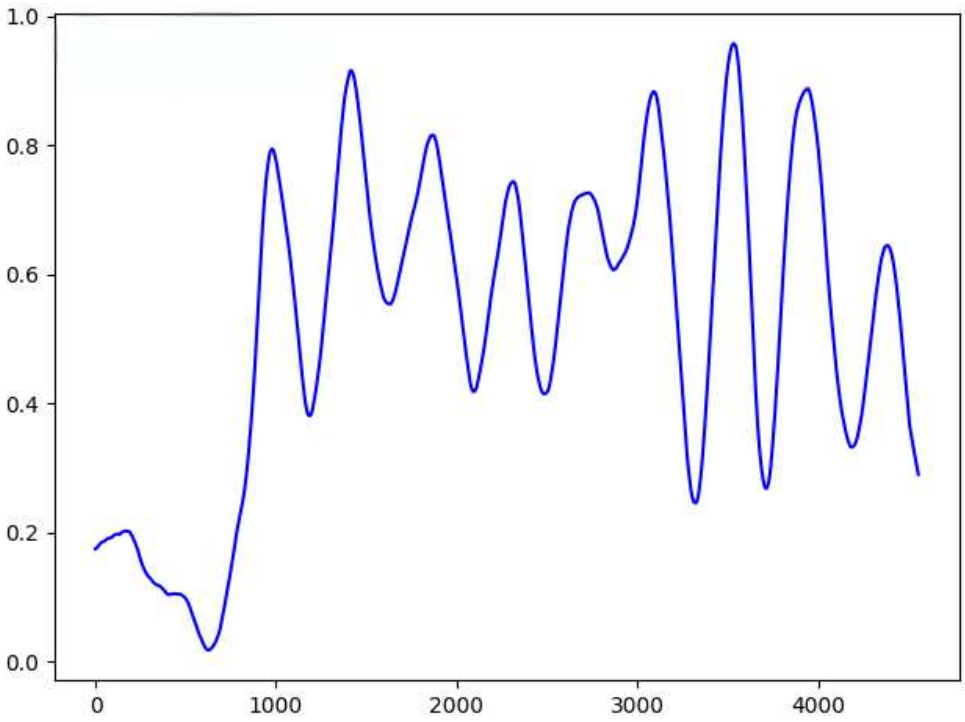


Figure 3.21: Smoothed model output

To automate the identification of these crucial points, the SciPy function `find_peaks`, as utilized in [50], is employed. Specifically designed for locating peaks in a signal, this function operates on a 1-D array and identifies all local maxima through simple comparisons of neighboring values.

In this study, the `find_peaks` function is configured with a key parameter known as `prominence`. The prominence of a peak is defined as the vertical distance between the peak and the start of the nearest slope. This parameter serves as a metric for evaluating how prominent a peak is within the signal. Fine-tuning the `prominence` enables the extraction of meaningful peaks, contributing to the robustness and accuracy of the automated identification process

In our context, setting an appropriate `prominence` threshold is crucial for accurate peak detection. After conducting several tests, we determined that a `prominence` value of 0.1 yields optimal results. This value strikes a balance between sensitivity and specificity, effectively capturing relevant peaks associated with repetition start, phase changes, and execution end. The automated nature of this process ensures efficiency and consistency in detecting key temporal markers, laying the groundwork for subsequent filtering and analysis.

In Figure 3.22, we can see the detected phase markers and the repetition markers detected by the function with the settings explained before.

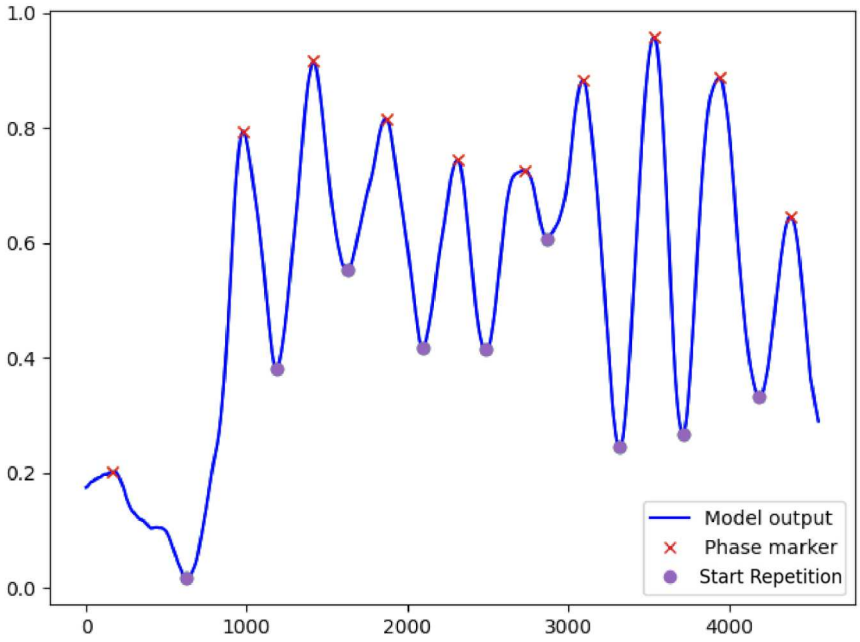


Figure 3.22: Markers detection

Force Repetition Pattern

The predicted sequence of start-phase-end markers may not consistently adhere to the expected repetition pattern throughout the entire predicted execution. To address this issue, a custom function has been developed to enforce the correct sequencing of these markers, ensuring the integrity of the identified repetitions.

Algorithm 1 impose_sequence

```
i ← 0
while i < len(results) − 1 do
  if results[i][1] == "phase" and results[i + 1][1] == "phase" then
    results.pop(i + 1)
  else if results[i][1] == "phase" and results[i + 1][1] == "rep" then
    results.insert(i + 1, (results[i + 1][0] + 10, "rep"))
    results.insert(i + 2, (results[i + 1][0] − 10, "rep"))
  else if results[i][1] == "rep" and results[i + 1][1] == "rep" then
    results.pop(i + 1)
  end if
  i ← i + 1
end while
if results[−1][1] == "phase" then
  results.pop(−1)
end if
if results and results[−1][1] == "rep" and results[−2][1] == "rep" then
  results ← results[: −1]
end if
if check_sequence(results) then
  return results
end if
return impose_sequence(results)
```

The custom function initiates by categorizing the detected peaks into start and end markers, discerning the initiation and conclusion points of each repetition. Subsequently, the peaks corresponding to phase transitions are examined. Inside each repetition sequence, there should ideally be a phase marker between each start/end marker pair, and no phase marker should occur between an end and a subsequent start marker.

The expected sequence for an execution with two repetitions, for instance, is as follows: start-phase-end-start-phase-end. The custom function meticulously evaluates the arrangement of phase markers within each repetition, choosing

only the most prominent phase marker if multiple are detected within a single repetition.

If there are too many phase markers within a repetition, only the most prominent one was retained, ensuring a focus on the salient phase transition. Conversely, if a repetition lacks a phase marker, the entire repetition was discarded from further analysis.

For example, in Figure 3.23, we can see the detected repetitions for a representative execution, where the blue line is the output model, and the red line represents the predicted ground truth (GT), with every triangle denoting a repetition and the peak of each triangle marking the phase-changing point. We can observe how the blue line presents a peak at the start of the execution but this is discarded even though a detected peak point was present at the start of the execution, visible in Figure 3.22. This was filtered by the custom function.

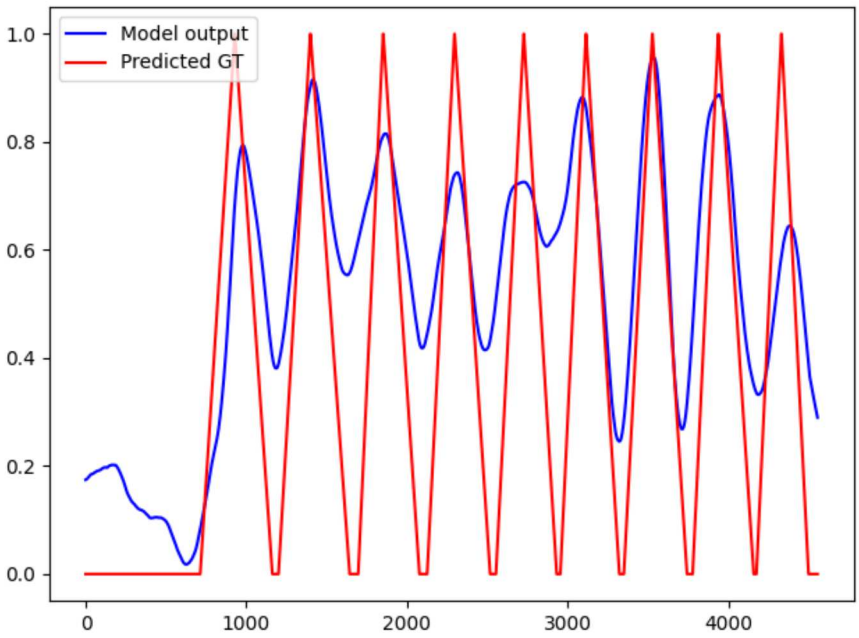


Figure 3.23: Predicted repetitions

This post-processing step, driven by the custom function, plays a key role in refining the model's predictions, enforcing consistency in the repetition pattern, and ultimately enhancing the reliability and interpretability of the results.

The post-processing phase serves as the critical bridge between raw model predictions and meaningful insights. By smoothing the output, detecting key

peaks, and enforcing a consistent repetition pattern, the predictions are refined and aligned with the expected temporal dynamics. This meticulous process enhances the reliability and interpretability of the model's output, setting the stage for a comprehensive evaluation of its performance.

3.6 Evaluation Metrics

The evaluation of machine learning models is a crucial aspect of the development process, providing insights into their performance and guiding further refinement. Proper evaluation metrics serve as the ground base for assessing how well a model generalizes to unseen data. Understanding the strengths and limitations of a model requires a comprehensive analysis of its predictions against ground truth, and this is where evaluation metrics play a pivotal role. In this section, we focus into the significance of these metrics in the context of our model for counting repetitions within exercise executions.

3.6.1 Counting-Based Evaluation Metrics

To quantify the performance of our model in counting repetitions within exercise executions, we introduce counting-based evaluation metrics. The primary objective is to determine the accuracy of the predicted repetitions concerning their alignment with the ground truth. The key criterion for a correct count is whether the predicted repetition is entirely contained within the corresponding real repetition.

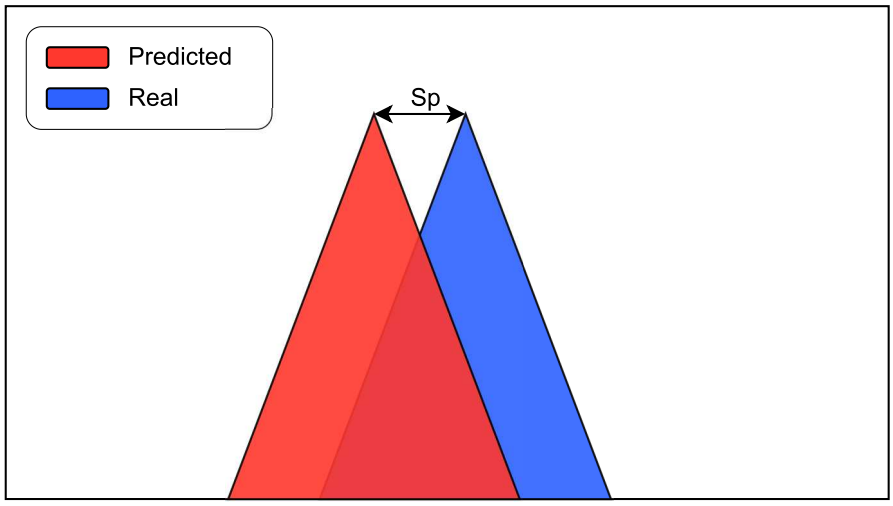


Figure 3.24: Correctly counted repetition resulting in True Positive

True Positives Definition

A predicted repetition is considered correct if either the start or the end of the predicted repetition falls within the bounds of the ground truth repetition.

This is illustrated in Figure 3.24, where the green shaded area represents correct counting.

False Positives and False Negatives Definition

If a predicted repetition is completely separated, without any common point, with the ground truth repetition, it is considered incorrect. Figure 3.25 illustrates an example of incorrect counting, where there is no pairing between predicted and real repetitions. Since there is a real repetition without a corresponding predicted repetition and a predicted repetition without a corresponding real repetition, these will count as one False Positive and one False Negative.

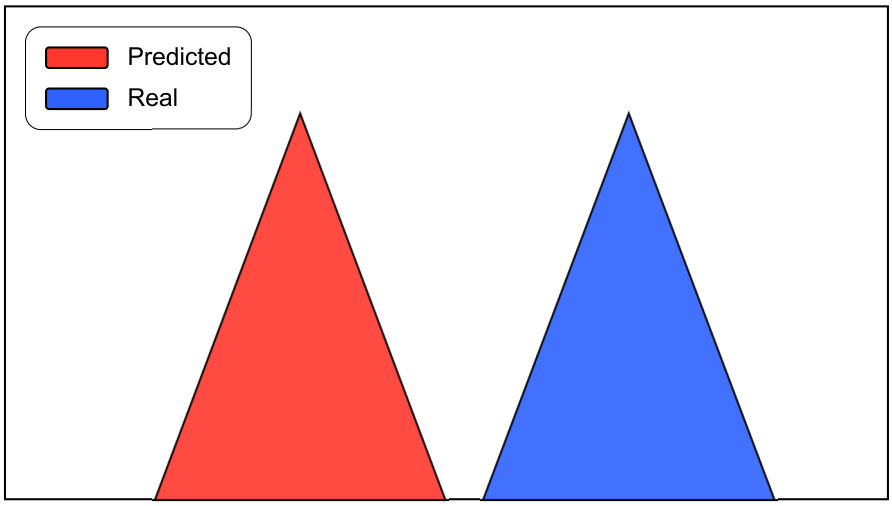


Figure 3.25: Incorrectly Counted Repetition resulting in a False Positive (red) and a False Negative (blue)

Handling Multiple Predictions

In cases where more than one predicted repetition is contained within a real repetition, only the predicted repetition with the lowest *shift_phase* is associated. This ensures that the model is credited for the prediction with the most accurate alignment with the ground truth.

Figure 3.26 provides an example where the first predicted repetition is associated with the real repetition due to its lower shift_phase (Sp1 < Sp2). In the example in Figure 3.26, there is one False Positive and one True Positive

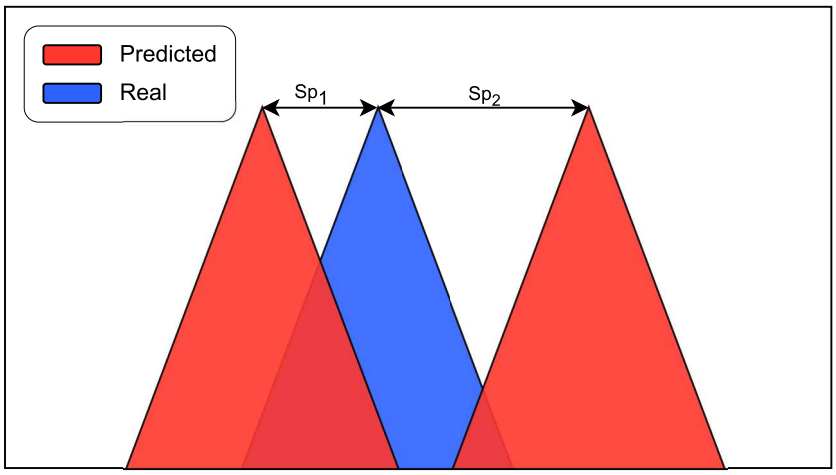


Figure 3.26: Handling Multiple Predictions

4

Results and Discussion

In this chapter, we focus on a comprehensive examination of the outcomes generated by our system in the time series segmentation. The assessment of the system's performance revolves around key metrics, providing insights into its predictive capabilities and the application in real-world scenarios.

The initial section of this chapter introduces the fundamental metrics employed for the evaluation of the system's performance, as outlined in Section 3.6.1. These metrics lay the groundwork for a rigorous analysis of the system's efficiency in detecting repetitions within IMU time series data.

Following this introduction, we transition to the presentation of results obtained through the implementation of the two distinct training methods The performance metrics, including True Positive Rate and False Discovery Rate are analyzed across a diverse set of exercises.

The subsequent segment of the chapter is a discussion of the system's performance. Each metric is examined in detail, emphasizing notable observations and insights gained from the experimental results. The chapter concludes with an overall discussion, consolidating the findings from each metric and offering a perspective on the system's performance. This comprehensive exploration not only presents the results obtained but also aims to provide a thorough understanding of the implications and limitations of the developed system, laying the way for future refinements and advancements in IMU time series segmentation within the field of tele-rehabilitation.

4.1 System metrics

In the assessment of the system's performance, the counting metrics explained in Section 3.6.1 were used. However, to obtain a more comprehensive overview of the system's capabilities across entire exercises or a set of exercises, we have devised general metrics.

These metrics are defined as follows:

True Positive Rate (TPR) also called sensitivity, is the probability that an actual positive will test positive. In this study this metrics represents the probability that the system detects a real repetition.

$$\begin{aligned} \text{True Positive Rate (\%)} &= \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \times 100 \\ &= \frac{\text{Correctly Predicted Repetitions}}{\text{Total } \textbf{Real} \text{ Repetitions}} \times 100 \end{aligned} \tag{4.1}$$

False Discovery Rate (FDR) represents the expected proportion of false discoveries among all hypotheses [51] [52]. In this research is the rate of overcounted repetitions over the total number of predicted repetitions.

$$\begin{aligned} \text{False Discovery Rate (\%)} &= \frac{\text{False Negatives}}{\text{True Positives} + \text{False Positives}} \times 100 \\ &= \frac{\text{Overcounted Repetitions}}{\text{Total } \textbf{Predicted} \text{ Repetitions}} \times 100 \end{aligned} \tag{4.2}$$

This set of metrics provides a general perspective on the system's performance, serving as a foundation for our subsequent comparative analysis and discussions.

4.2 Results

The system was tested on the test set, the data in the test set was not used in the model training in order to give a fair evaluation of the overall system, the test set consists in a total of 494 executions divided for the exercises.

True Positive Rate The True Positive Rate on the test set for each exercise is illustrated in Figure 4.1. The blue columns represent the True Positive Rate for the Baseline training approach called *Baseline* approach, while the orange lines show the results for the *Fine Tuning* approach.

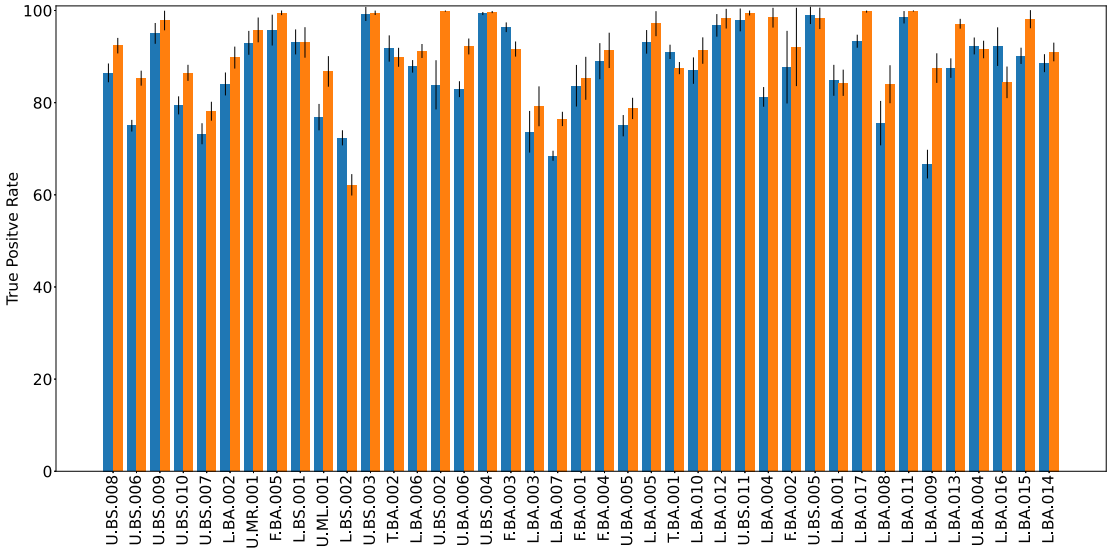


Figure 4.1: True Positive Rate comparison between Baseline and Fine Tuning approach

For a closer examination of the differences, a zoomed-in version of the plot is presented in Figure 4.2. Notably, the Fine Tuning approach has 25 out of 41 exercises with a TPR over 90%, compared to 18 exercises with the Baseline approach. However, it is worth highlighting that the exercise **L.BS.002**, "Elevation on the forefoot" in Tab. 3.4 registers the lowest TPR. This exercise, demanding strength and the ability to maintain a specific position for a brief duration, creates challenges in terms of noise and stability in recorded data, potentially contributing to its comparatively lower performance.

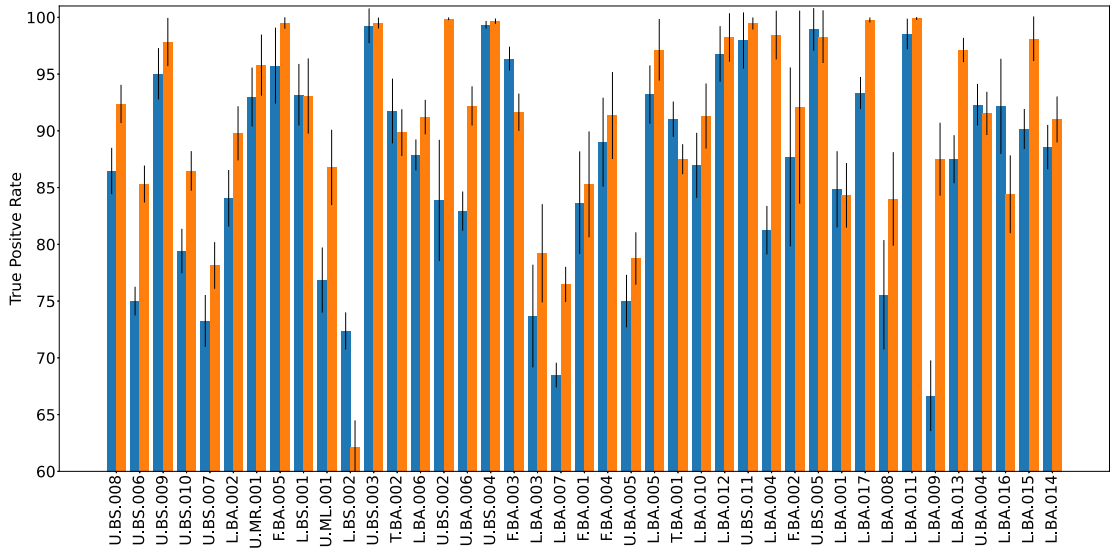


Figure 4.2: True Positive Rate comparison between Baseline and Fine Tuning approach above 60%

In Figure 4.3, we can observe the True Positive Rate difference for each exercise resulting from adopting the Fine Tuning approach compared to the Baseline approach. The bars are colored green when the True Positive Rate increases, indicating an improvement in performance. We can see that 32 exercises over 41 have a increase in TPR with the Fine Tuning approach compared to the Baseline approach.

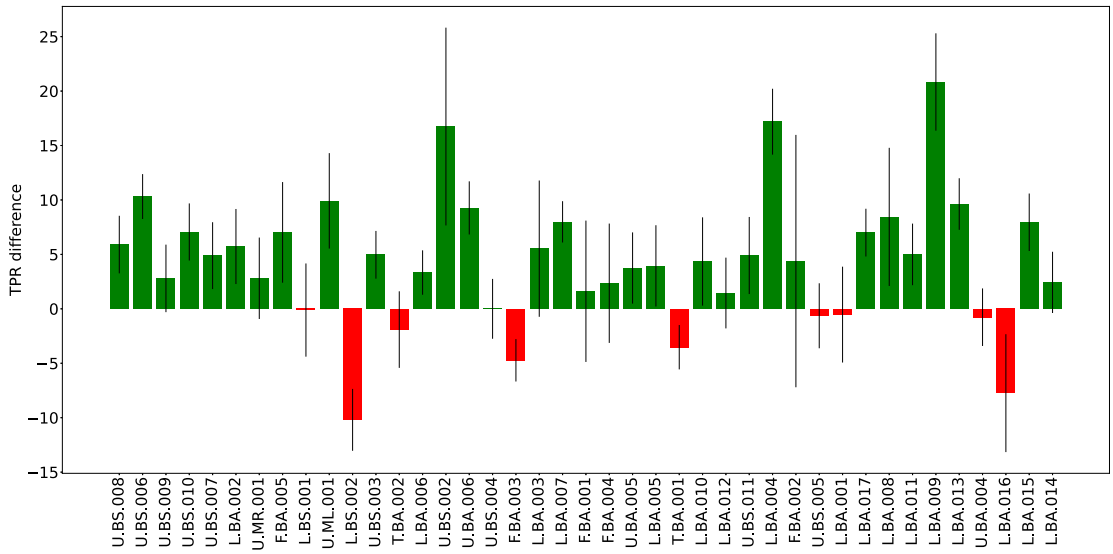


Figure 4.3: True Positive Rate difference between Baseline and Fine Tuning approach

In Figure 4.1, it is visible that both approaches yield satisfactory results, with the average TPR over the entire test set of around 82% for the Baseline Approach and 86% for the Fine Tuning approach. However, the difference in results between the two approaches is well visible in Figure 4.3. Almost every exercise performs better with the Fine Tuning approach, with multiple exercises showing improvements exceeding 10%. However, there are exercises where the TPR remains the same, and 9 exercises over 41 exhibit a lower TPR.

False Discovery Rate Figure 4.4 displays the False Discovery Rate on the test set for each exercise. The blue columns correspond to the FDR for the Baseline approach, while the orange columns represent the Fine Tuning approach.

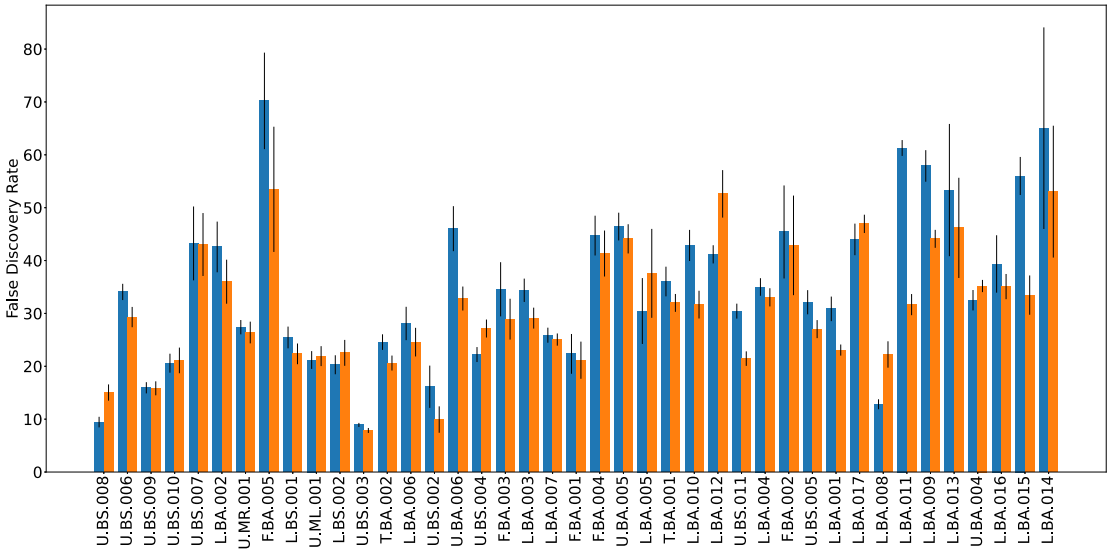


Figure 4.4: False Discovery Rate comparison between Baseline and Fine Tuning approach zoomed

The average False Discovery Rate across the entire test set stands at approximately 31% for the Baseline approach and is slightly lower, at 27%, for the Fine Tuning approach. Although some exercises exhibit an FDR surpassing 40%, with instances even reaching up to 70%, it's noteworthy that 26 out of 41 exercises maintain an FDR below the 30% threshold under the Fine Tuning approach. This indicates that there is a 27% likelihood that any predicted repetition might not accurately correspond to an actual repetition. The overall FDR results are considered to be less than optimal. The exercise with the highest FDR is **F.BA.005** "Marinating balance in single support", which achieves a TPR of 100% but exhibits a FDR of approximately 70% for the Baseline approach

and around 50% for the Fine Tuning approach. This suggests that the system is overly sensitive and predicts numerous repetitions even when not warranted. The nature of the exercise, requiring balance and strength with the patient maintaining equilibrium using only one leg (as detailed in Table 3.1), contributes to a high variance in repetition duration, as depicted in Figure 3.11. Additionally, this exercise has a limited number of executions, as illustrated in Figure 3.5 that can suggest that the model does not have enough data to learn that repetition pattern. It's worth mentioning that with the Fine Tuning approach this exercise has both a higher TPR and a lower FDR with respect to the Baseline approach.

Figure 4.5 illustrates the False Discovery Rate difference, indicating the impact of transitioning from the Baseline to the Fine Tuning approach. Green bars signify a reduction in FDR, reflecting an enhancement in precision. 31 exercises over 41 have a lower FDR with the Fine tuning approach, with 7 of this exercises exhibiting a difference of more than 10%.

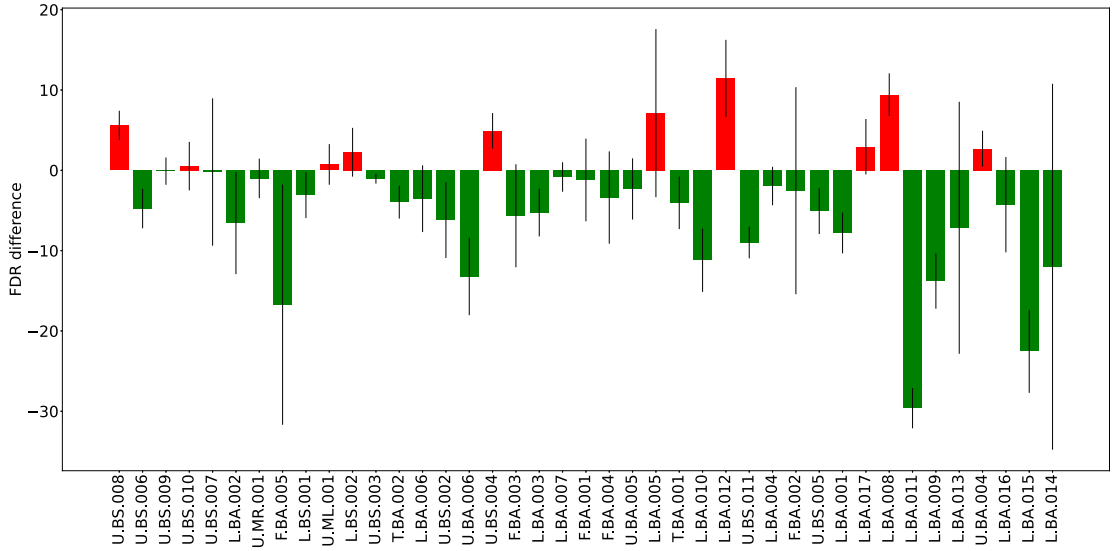


Figure 4.5: False Discovery Rate difference between Baseline and Fine Tuning approach

Comparative analysis of the True Positive Rate and False Discovery Rate between the Fine Tuning and Baseline approaches reveals a clear superiority of the former for the majority of exercises. Specifically, 24 out of 41 exercises exhibited an improvement in TPR while simultaneously experiencing a reduction in FDR when subjected to the Fine Tuning approach.

Figure 4.6 illustrates a comprehensive comparison between the outcomes of the Baseline approach (represented by the dashed line) and the Fine Tuning approach (depicted by the solid line) for both metrics. The graphical representation confirms that, across a majority of exercises, the Fine Tuning approach consistently surpasses the performance of the Baseline model. Notably, the performance difference is particularly pronounced in several exercises. Future research could delve into exploring the potential relationships among exercises that exhibit substantial benefits from the Fine Tuning approach compared to the Baseline. Understanding these relationships could provide valuable insights into the specific characteristics or patterns that contribute to the enhanced performance observed with the Fine Tuning methodology.

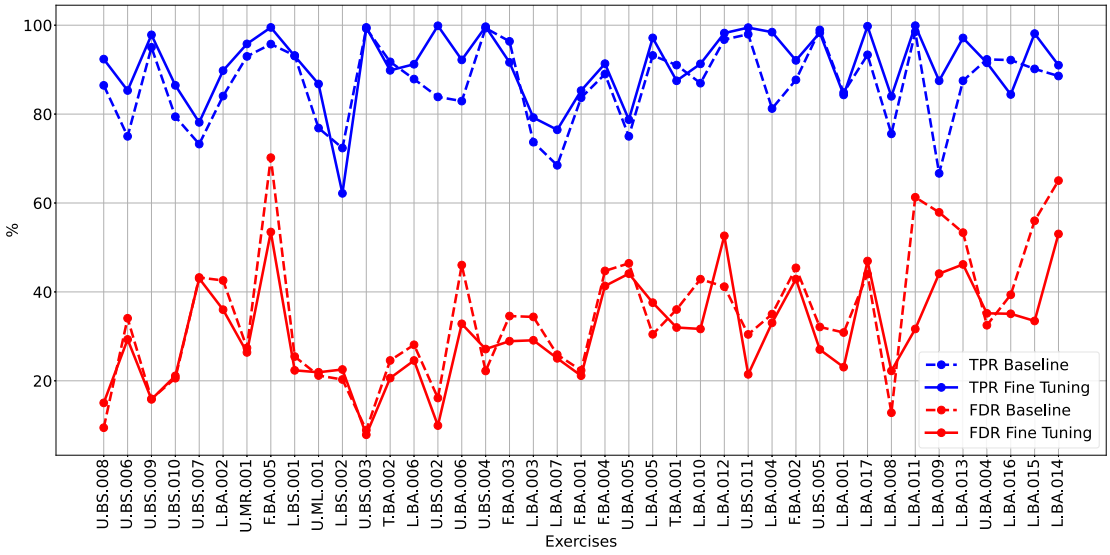


Figure 4.6: True Positive Rate and False Discovery Rate for the Baseline approach vs the Fine Funing approach

However, there are exceptions. Only two exercises demonstrated better performance across both metrics with the Baseline approach: **U.BA.004** "Seated arm extension" and **L.BS.002** "Elevation on the forefoot," as both detailed in Table 3.4. Interestingly, these two exercises do not share apparent similarities; the former involves upper body movement, while the latter focuses on the lower body. They are classified into distinct groups and there the volume of data available for each exercise differs substantially, with **U.BA.004** having 173 executions compared to **L.BS.002**'s 82.

One noteworthy observation is the exemplary performance of the exercise **U.BS.003** "Elevation of both arms with a stick while standing". This exercise boasts a TPR of 100%, and an FDR metric close to 0%. The system predicts each repetition almost flawlessly. The effectiveness of this exercise may be attributed to its relatively simple movement with the use of a stick aids in coordinating the movements of both arms, as indicated in Table 3.6, coupled with a substantial number of executions, as evident in Figure 3.5.

It's crucial to note that several challenges and variations exist within the dataset. Moreover, patients exhibit diverse approaches to exercises. For instance, some patients may follow a different sequence of movements for the same exercise. These variations introduce complexity and highlight the importance of robustness in the system's performance under real-world conditions. Also, the results also show one of the challenges of this research, the high number of exercises, this introduce a big disparity in the results with an high variance in both TPR and FDR metrics between the exercises.

Overall, the results demonstrate that the system performs better with the Fine Tuning approach, considering both metrics employed. The TPR metric exhibits high accuracy showcasing satisfactory performance. However, there is room for improvement in the FDR metric, which, while acceptable, could be optimized further.

5

Conclusions

The study introduces a deep learning model designed for the segmentation of IMU time series in the context of tele-rehabilitation. The dataset, acquired through the ARC Intellicare producted by Henesis s.r.l., includes data from over 62 participants (both healthy and pathological) engaged in a comprehensive set of 41 rehabilitation exercises, captured using three IMU sensors located on the trunk, on the wrists and on the ankles.

The results revealed a superior performance of the fine-tuning approach, involving the training of a group of exercises aggregated based on their movement duration similarity, followed by fine-tuning the models for each exercise. This approach outperforms the baseline method, which involves training each exercise individually with only its specific data. The fine-tuning approach achieved a True Positive Rate of over 90% for the majority of exercises (25 out of 41) within the test set.

However, there is still room for improvement in terms of the False Discovery Rate, possibly attributed to the complexity of the dataset, featuring a high number of exercises never addressed in existing literature on IMU time series segmentation for tele-rehabilitation.

The results also revealed significant variations in performance among different exercises. This variability can be attributed to the diverse nature of movements required for each exercise, with some emphasizing the mobility of the upper limbs, others focusing on the lower limbs, and some involving the entire body.

Future research could explore alternative model architectures to increase

the segmentation accuracy, investigate additional post-processing techniques to enhance the model output, and explore other evaluation metrics to have a better understanding of the performances.

This work, with respect to similar studies reported in Section 2, investigated a set of movements significantly larger than those explored in the current state of the art. The complexity introduced by the broad range of movements in the dataset is a distinctive aspect of this study, aiming to fill a notable gap in the literature regarding IMU time series segmentation in the field of tele-rehabilitation.

In conclusion, this study successfully developed a deep learning solution for the detection of repetitions in specific tele-rehabilitation exercises using IMU sensors, laying the groundwork for further advancements in this evolving field, as well as contributing in the technological improvement of the company's product.

References

[1] Marcelo Dalbosco-Salas et al. "Effectiveness of a primary care telerehabilitation program for post-COVID-19 patients: a feasibility study". In: *Journal of clinical medicine* 10.19 (2021), p. 4428.

[2] Dimitrios-Sokratis Komaris et al. "Unsupervised IMU-based evaluation of at-home exercise programmes: a feasibility study". In: *BMC Sports Science, Medicine and Rehabilitation* 14.1 (2022), p. 28.

[3] Ali Barzegar Khanghah, Geoff Fernie, and Atena Roshan Fekr. "Design and Validation of Vision-Based Exercise Biofeedback for Tele-Rehabilitation". In: *Sensors* 23.3 (2023), p. 1206.

[4] Bendong Zhao et al. "Convolutional neural networks for time series classification". In: *Journal of Systems Engineering and Electronics* 28.1 (2017), pp. 162–169.

[5] "Assisted rehabilitation care during post-stroke mANaGement: fEasibiLity assessment (ARCANGEL)". In: *ClinicalTrials.gov identifier: NCT03787433* (Accessed: January 16, 2024.). URL: https://clinicaltrials.gov/study/NCT03787433.

[6] Marianna Capecci et al. "Telerehabilitation with ARC Intellicare to Cope with Motor and Respiratory Disabilities: Results about the Process, Usability, and Clinical Effect of the ldquo;Ricominciarerdquo; Pilot Study". In: *Sensors* 23.16 (2023). ISSN: 1424-8220. DOI: 10.3390/s23167238. URL: https://www.mdpi.com/1424-8220/23/16/7238.

[7] Mauro Zampolini et al. "Tele-rehabilitation: Present and future". In: *Annali dell'Istituto superiore di sanità* 44 (Jan. 2008), pp. 125–34.

[8] Ines Frederix et al. "A review of telerehabilitation for cardiac patients". In: *Journal of Telemedicine and Telecare* 21.1 (2015). PMID: 25475219, pp. 45–53. DOI: 10.1177/1357633X14562732. URL: https://doi.org/10.1177/1357633X14562732.

[9] Kate E Laver et al. "Telerehabilitation services for stroke". In: *Cochrane Database of Systematic Reviews* 1 (2020).

[10] Janet Prvu Bettger and Linda J Resnik. "Telerehabilitation in the Age of COVID-19: An Opportunity for Learning Health System Research". In: *Physical Therapy* 100.11 (Aug. 2020), pp. 1913–1916. ISSN: 1538-6724. DOI: 10.1093/ptj/pzaa151. eprint: https://academic.oup.com/ptj/article-pdf/100/11/1913/48748642/pzaa151.pdf. URL: https://doi.org/10.1093/ptj/pzaa151.

[11] Pamela Seron et al. "Effectiveness of Telerehabilitation in Physical Therapy: A Rapid Overview". In: *Physical Therapy* 101.6 (Feb. 2021), pzab053. ISSN: 1538-6724. DOI: 10.1093/ptj/pzab053. eprint: https://academic.oup.com/ptj/article-pdf/101/6/pzab053/48728453/pzab053.pdf. URL: https://doi.org/10.1093/ptj/pzab053.

[12] David Hailey et al. "Evidence of benefit from telerehabilitation in routine care: a systematic review". In: *Journal of Telemedicine and Telecare* 17.6 (2011). PMID: 21844172, pp. 281–287. DOI: 10.1258/jtt.2011.101208. eprint: https://doi.org/10.1258/jtt.2011.101208. URL: https://doi.org/10.1258/jtt.2011.101208.

[13] Miia M Jansson et al. "The effects and safety of telerehabilitation in patients with lower-limb joint replacement: A systematic review and narrative synthesis". In: *Journal of Telemedicine and Telecare* 28.2 (2022). PMID: 32316837, pp. 96–114. DOI: 10.1177/1357633X20917868. eprint: https://doi.org/10.1177/1357633X20917868. URL: https://doi.org/10.1177/1357633X20917868.

[14] Fred S. Sarfo et al. "Tele-Rehabilitation after Stroke: An Updated Systematic Review of the Literature". In: *Journal of Stroke and Cerebrovascular Diseases* 27.9 (2018), pp. 2306–2318. ISSN: 1052-3057. DOI: https://doi.org/10.1016/j.jstrokecerebrovasdis.2018.05.013. URL: https://www.sciencedirect.com/science/article/pii/S1052305718302313.

[15] Giuseppe Placidi. "A smart virtual glove for the hand telerehabilitation". In: *Computers in Biology and Medicine* 37.8 (2007), pp. 1100–1107. ISSN: 0010-4825. DOI: https://doi.org/10.1016/j.compbiomed.2006.09.011. URL: https://www.sciencedirect.com/science/article/pii/S0010482506001776.

[16] Varleisha Gibbs and Susan Toth-Cohen. "Family-Centered Occupational Therapy and Telerehabilitation for Children with Autism Spectrum Disorders". In: *Occupational Therapy In Health Care* 25.4 (2011). PMID: 23899082, pp. 298–314. DOI: 10.3109/07380577.2011.606460. eprint: https://doi.org/10.3109/07380577.2011.606460. URL: https://doi.org/10.3109/07380577.2011.606460.

[17] Alessandra Berton et al. "Virtual Reality, Augmented Reality, Gamification, and Telerehabilitation: Psychological Impact on Orthopedic Patients Rehabilitation". In: *Journal of Clinical Medicine* 9.8 (2020). ISSN: 2077-0383. DOI: 10.3390/jcm9082567. URL: https://www.mdpi.com/2077-0383/9/8/2567.

[18] Jie Hao et al. "Effects of virtual reality-based telerehabilitation for stroke patients: A systematic review and meta-analysis of randomized controlled trials". In: *Journal of Stroke and Cerebrovascular Diseases* 32.3 (2023), p. 106960. ISSN: 1052-3057. DOI: https://doi.org/10.1016/j.jstrokecerebrovasdis.2022.106960. URL: https://www.sciencedirect.com/science/article/pii/S1052305722006516.

[19] Amin M. Nasrabadi et al. "A new scheme for the development of IMU-based activity recognition systems for telerehabilitation". In: *Medical Engineering Physics* 108 (2022), p. 103876. ISSN: 1350-4533. DOI: https://doi.org/10.1016/j.medengphy.2022.103876. URL: https://www.sciencedirect.com/science/article/pii/S1350453322001242.

[20] W. Kong et al. "Development of a real-time IMU-based motion capture system for gait rehabilitation". In: (2013), pp. 2100–2105. DOI: 10.1109/ROBIO.2013.6739779.

[21] Khimraj et al. "Human Activity Recognition using Accelerometer and Gyroscope Data from Smartphones". In: (2020), pp. 1–6. DOI: 10.1109/ICONC345789.2020.9117456.

[22] Matteo Gadaleta et al. "Deep Learning Techniques for Improving Digital Gait Segmentation". In: *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. 2019, pp. 1834–1837. DOI: 10.1109/EMBC.2019.8856685.

[23] Ilham Faisal, Tito Purboyo, and Anton Ansori. "A Review of Accelerometer Sensor and Gyroscope Sensor in IMU Sensors on Motion Capture". In: *Journal of Engineering and Applied Sciences* 15 (Nov. 2019), pp. 826–829. DOI: 10.36478/jeasci.2020.826.829.

[24] Lin Zhou et al. "How We Found Our IMU: Guidelines to IMU Selection and a Comparison of Seven IMUs for Pervasive Healthcare Applications". In: *Sensors* 20.15 (2020). ISSN: 1424-8220. DOI: 10.3390/s20154090. URL: https://www.mdpi.com/1424-8220/20/15/4090.

[25] Stefano Paolucci Marco Iosa Pietro Picerno and Giovanni Morone. "Wearable inertial sensors for human movement analysis". In: *Expert Review of Medical Devices* 13.7 (2016), pp. 641–659. DOI: 10.1080/17434440.2016.1198694. URL: https://doi.org/10.1080/17434440.2016.1198694.

[26] Miodrag Lovri, Marina Milanovi, and Milan Stamenkovi. "Algoritmic methods for segmentation of time series: An overview". In: *Journal of Contemporary Economic and Business Issues* 1.1 (2014), pp. 31–53.

[27] Kim Myeongsub and Park Sukyung. "Golf Swing Segmentation from a Single IMU Using Machine Learning". In: *Sensors* 20.16 (2020). ISSN: 1424-8220. DOI: 10.3390/s20164466. URL: https://www.mdpi.com/1424-8220/20/16/4466.

[28] Luktuke Yadnyesh Y. and Hoover Adam. "Segmentation and Recognition of Eating Gestures from Wrist Motion using Deep Learning". In: (2020), pp. 1368–1373. DOI: 10.1109/BigData50022.2020.9378382.

[29] Shun Ishii et al. "ExerSense: Real-Tme Physical Exercise Segmentation, Classification, and Counting Algorithm Using an IMU Sensor". In: (2020). arXiv: 2004.10026 [cs.HC].

[30] "La ripresa della attività dopo COVID-19". In: (Accessed: January 16, 2024), p. 14. URL: https://www.rehab-univpm.it/public/assets/img/home/booklet_covid.pdf.

[31] "Arc intellicare clinical tial Clinical Trial". In: *ClinicalTrials.gov Identifier: NCT05074771* (Accessed: January 17, 2024). URL: https://clinicaltrials.gov/study/NCT05074771.

[32] Dalwinder Singh and Birmohan Singh. "Investigating the impact of data normalization on classification performance". In: *Applied Soft Computing* 97 (2020), p. 105524. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2019.105524. URL: https://www.sciencedirect.com/science/article/pii/S1568494619302947.

[33] Chia-Shang James Chu. "Time series segmentation: A sliding window approach". In: *Information Sciences* 85.1 (1995), pp. 147–173. ISSN: 0020-0255. DOI: https://doi.org/10.1016/0020-0255(95)00021-G. URL: https://www.sciencedirect.com/science/article/pii/002002559500021G.

[34] Sojeong Ha and Seungjin Choi. "Convolutional neural networks for human activity recognition using multiple accelerometer and gyroscope sensors". In: *2016 International Joint Conference on Neural Networks (IJCNN)* (2016), pp. 381–388. URL: https://api.semanticscholar.org/CorpusID:2281930.

[35] John Cristian Borges Gamboa. "Deep Learning for Time-Series Analysis". In: (2017). arXiv: 1701.01887 [cs.LG].

[36] Lamyaa Sadouk. "CNN approaches for time series classification". In: *Time series analysis-data, methods, and applications* 5 (2019).

[37] Hao Zhou, José Manuel Álvarez, and Fatih Murat Porikli. "Less Is More: Towards Compact CNNs". In: *European Conference on Computer Vision*. 2016. URL: https://api.semanticscholar.org/CorpusID:14666124.

[38] Keiron O'Shea and Ryan Nash. *An Introduction to Convolutional Neural Networks*. 2015. arXiv: 1511.08458 [cs.NE].

[39] Deqing Sun et al. "A fully-connected layered model of foreground and background flow". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2013, pp. 2451–2458.

[40] S.H. Shabbeer Basha et al. "Impact of fully connected layers on performance of convolutional neural networks for image classification". In: *Neurocomputing* 378 (2020), pp. 112–119. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2019.10.008. URL: https://www.sciencedirect.com/science/article/pii/S0925231219313803.

[41] Stefan Wager, Sida Wang, and Percy S Liang. "Dropout Training as Adaptive Regularization". In: 26 (2013). Ed. by C.J. Burges et al. URL: https://proceedings.neurips.cc/paper_files/paper/2013/file/38db3aed.pdf.

[42] Pierre Baldi and Peter J Sadowski. "Understanding dropout". In: *Advances in neural information processing systems* 26 (2013).

[43] Wei-Han Lee et al. *Time Series Segmentation through Automatic Feature Learning*. 2018. arXiv: 1801.05394 [cs.LG].

[44] David Guijo-Rubio et al. "Time-Series Clustering Based on the Characterization of Segment Typologies". In: *IEEE Transactions on Cybernetics* 51.11 (2021), pp. 5409–5422. DOI: 10.1109/TCYB.2019.2962584.

[45] Qiong Cao et al. "Greenhouse Temperature Prediction Based on Time-Series Features and LightGBM". In: *Applied Sciences* 13.3 (2023). ISSN: 2076-3417. DOI: 10.3390/app13031610. URL: https://www.mdpi.com/2076-3417/13/3/1610.

[46] Giulia Cisotto et al. "hvEEGNet: exploiting hierarchical VAEs on EEG data for neuroscience applications". In: (2023).

[47] Leslie N. Smith. "No More Pesky Learning Rate Guessing Games". In: *CoRR* abs/1506.01186 (2015). arXiv: 1506.01186. URL: http://arxiv.org/abs/1506.01186.

[48] Diederik Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: (2015).

[49] Frank Nielsen and Frank Nielsen. "Hierarchical clustering". In: *Introduction to HPC with MPI for Data Science* (2016), pp. 195–211.

[50] Joshua Sun et al. "Amplitude and Frequency Based Evaluations for Algorithm Development of Premature Ventricular Contraction Detection System". In: *2022 4th International Conference on Cybernetics and Intelligent System (ICORIS)*. IEEE. 2022, pp. 1–4.

[51] C.R. Genovese. "False Discovery Rate Control". In: *Brain Mapping*. Ed. by Arthur W. Toga. Waltham: Academic Press, 2015, pp. 501–507. ISBN: 978-0-12-397316-0. DOI: https://doi.org/10.1016/B978-0-12-397025-1.00323-7. URL: https://www.sciencedirect.com/science/article/pii/B9780123970251003237.

[52] Yoav Benjamini and Yosef Hochberg. “Controlling the false discovery rate: a practical and powerful approach to multiple testing”. In: *Journal of the Royal statistical society: series B (Methodological)* 57.1 (1995), pp. 289–300.