POLITECNICO DI MILANO

Master of Science in Biomedical Engineering

Department of Electronics, Information and Bioengineering

BIOMECHANICAL MODEL AND MACHINE LEARNING ALGORITHMS COMPARISON FOR CUSTOMIZED TRAINING BIOFEEDBACK ON ISS

Supervisor: Prof. Giancarlo Ferrigno

Co - supervisor: Ing. Martina Ravizza

Master Thesis of:

Mattia Quaranta Student ID: 919661

Marco Sinatra Student ID: 920039

Academic year 2020/2021

ACKNOWLEDGEMENT

The first special thank goes to the engineer Martina Ravizza who followed us with constancy and dedication every week during the development of this project.

We thank Professor Alessandra Pedrocchi for advising us to contact and collaborate with Professor Giancarlo Ferrigno for this thesis opportunity.

Also, we cannot help thanking Sir Danilo Catelli since he tried to help us with OpenSim even though his support did not become fundamental to reach our aims.

One last expression of gratitude goes to Laura Giani who was carrying out another parallel thesis (included in the MARS-PRE project) and provided us with the raw acceleration data of the subjects considered in our work.

ABSTRACT

State of art and aims

The interest in space exploration of the past 60 years has proven how humanity always tries to reach the impossible. Even though studying another planet different from Earth seemed unfeasible, nowadays the possibility of landing on Mars is becoming a concrete reality. However, such ambitious project comes together with unpleasant downsides, in terms of human's health and adaptation in a microgravity environment. Of all the drawbacks while living in a place lacking gravity, we could list cardiovascular, respiratory, visual and musculoskeletal issues. Among these, for instance, many side effects regard Space Adaptation Syndrome, uneven ventilation, irregular perfusion, muscle deconditioning, atrophy and bone demineralization. For research purposes, a lot of projects tried to replicate and study the same specific environmental conditions found by the space travelers. Initially, the immersion in water was thought to be a logical model for reducing the pull of gravity on the body mass, but it soon resulted an impractical and inefficient method. Bed rest, whose longevity predates space exploration, later became a practical application to simulate the weightlessness conditions on ground. With the progress of Computer Science, also simulations via software (e.g., OpenSim) became a fundamental pillar in this context.

To prevent deconditioning in future long-duration missions (LDMs), crewmembers still follow a specific training protocol pre-, in- and post-flight. Indeed, NASA is using the International Space Station (ISS) to learn more about physiological countermeasures. Both cardiovascular and resistive training are performed with specific exercise devices: CEVIS (Cycle Ergometer with Vibration Isolation and

Stabilization) and treadmill COLBERT are exploited for cardiovascular system health; the Advance Resistive Exercise Device (ARED) is used, instead, to prevent muscle atrophy and bone mineral loss. It simulates the use of free weights in microgravity by generating a constant load, which can be changed from 0 to 272.5 Kg. ARED allows to carry out up to 29 exercises including Normal Stance Squat, Wide Stance Squat and Deadlift. This mechanical architecture is also combined with a brand-new motion capture system (MOCAP), namely BTS SMART-DX. The latter is going to collect kinematic data basing on passive markers placed on the cosmonauts' body.

This thesis is inserted in both research projects "MARcatori biologici e funzionali per la biomedicina aStronautica di PREcisione – MARS-PRE", proposed by Italian Space Agency (ASI) and "ARED Kinematics – Biomechanical quantification of bone and muscle loading to improve the quality of microgravity countermeasure prescription for resistive exercise", which involves European Space Agency (ESA), Neuroengineering and Medical Robotics Laboratory (NearLab) of Politecnico di Milano, Johnson Space Center (JSC) of National Aeronautics and Space Administration (NASA) and Kayser Italia. The first project aims to find biological and functional anticipatory makers of musculo-skeletal damage, which can occur while performing inaccurately the daily workout routine on ISS. Then, a real time system will interact and provide information/warnings on correct/incorrect performance execution. The second project, ARED-K, plans on improving the subject specific effectiveness of daily exercises on flight by estimating internal body loads.
To reach these purposes, data collection using motion capture system and force plates during pre-, in- and post-flight is needed, as well as biomechanical analysis and statistical comparison of data collected.

Currently, no inertial sensor data of exercises performed with ARED are available, but only kinematic ones (obtained by using the MOCAP system). For this reason, a

biomechanical model is needed to simulate sensors placement on body and extract analogues data.

The present work hence focused on the following aims: (1) improvement of a model useful to conduct a biomechanical analysis of the human motion in microgravity conditions on ISS; (2) simulation of inertial sensors in different body points, thus obtaining acceleration signals related to correct and incorrect exercise executions; (3) the realization of a system based on machine learning classification that will interact and provide information/warnings on correct/incorrect performance during typical workout routine (e.g. when performing Normal Squat, Wide Squat and Deadlift); (4) Once chosen the most suitable ML algorithm, validation of a customized and subject-specific classifier to recognize correct/incorrect exercise executions.

Inertial sensors simulation, IMU data and experimental protocol

OpenSim is the open-source software exploited to conduct the biomechanical analysis and the microgravity simulations. The platform allows, through to the "Analyze" tool, to simulate sensors placement on specific body points. The body points chosen were Sternum, Sacrum, Right Upper Thigh, Left Upper Thigh, Right Upper Shank and Left Upper Shank. In this way, simulated acceleration data can be extracted. The musculoskeletal model generally used for gait and running tasks is "Gait2392" but is not suitable for deep squatting exercises. In literature, the already validated "Catelli" model demonstrated to increase the ROMs especially for the knee joint, thus implying its usability for the motions considered in the current project. In particular, this model has been validated specifically for Squat exercises, which require higher ROMs for the knee joint with respect to gait and running. We also decided to verify the "Catelli" model validation of the research paper, successfully. To reach this purpose, we considered the simulated data coming from 6 subjects (3 Males, Mean: 32.33 ± 3.06 years old, 79.96 ± 10.84 kg, 174.67 ± 8,50 cm height; 3 Females, Mean: 31.33 ± 6.80 years old, 60.20 ± 7.56 kg, 161.83 ± 7.21 cm height) recruited at NASA JSC who performed 4 repetitions for each movement (Normal Squat, Wide Squat and Deadlift) using ARED.

As regards the real accelerations acquired through IMU sensors, instead, we started from the raw data (collected in Pavia) of another parallel thesis project inserted in the MARS-PRE research. From the work of Ravizza et al. (2020), no statistical differences are observed between kinematics of exercises performed with ARED and barbell. Consequently, the current thesis project relied on this pillar to conduct the acquisition step. The Electronic system used is made up of 6 different accelerometers (XSENS) placed in specific human body locations (1 on each thigh, 1 on each shank, 1 on sternum and 1 on sacrum). Only 17 subjects were involved (9 Males, Mean: 26.89 ± 5.73 years old, 64.22 ± 7.14 kg, 173.44 ± 4.25 cm height; 8 Females, Mean: 25.38 ± 3.77 years old, 56 ± 6.31 kg, 163,14 ± 6.52 cm height). They performed a different number of repetitions (the range varies from 5 to 20, according to their physical capabilities) of Normal Squat, Wide Squat and Deadlift (similarly to those executed at NASA JSC) using barbell and weights (load within 50-75% of ISO-MAX). These collected data were not simply referred to correct exercise execution, but also to a dataset of incorrect ones (always avoiding harming permanently or temporarily the physiological structures of the subjects recruited) which are approved by specialists of NASA JSC and by the Etic Committee of Politecnico di Milano.

Data processing and classifier development

Matlab R2019b was used to pre-process the IMU acceleration data. Raw data were initially filtered with a sixth-order Butterworth low pass filter, with a cut-off frequency of 5 Hz. After the removal of outliers and the extrapolation of only the meaningful data, every single exercise repetition was isolated and resized to an average value. These steps allow a quicker usage for the next part: Google Colab (based on python scripts) was employed to normalize, apply an algorithm able to solve imbalance classification issues and, eventually, to perform feature extraction and selection (PCA). The latter enables to reduce the feature set and it is applied to all the algorithms explored in this study (e.g., SVM, KNN, Decision Tree, XGBoost, MLP) except for the Convolutional Neural Network (CNN), which simply employed the already pre-processed data. These supervised learning approaches were developed and tested to perform a multi-label classification, in order to discriminate

correct and incorrect exercises (5 and 3 different inaccurate motions were classified for Normal/Wide Squat and Deadlift, respectively).

Results and conclusions

Biomechanical model suitable for ISS training exercise - The "Catelli" OpenSim biomechanical model, differently from "gait2392", demonstrated to be a valid candidate in the ARED-K research project since it allows to reach, especially for the Knee joint, those Range of Motions (ROM) suitable for the specific training exercises (Normal Squat, Wide Squat, Deadlift) on ISS.

Performance classification based on acceleration signals - The original dataset, containing 1314 features for six sensors (hence, 219 features for each of them), is reduced with PCA to 198, 216 and 152 features for Normal Squat, Wide Squat and Deadlift, respectively (thus describing at least 90% of data variance). SVM and CNN allowed, respectively, to reach discrete and reliable results in terms of accuracy: 87.79% and 84.30% for Normal Squat; 86.38% and 86.91 % for Wide Squat; 83.76% and 82.05% for Deadlift. Further investigations are needed to enlarge the dataset and refine the classifier, but these preliminary results could be seen as an incentive to consider this approach a working solution. We strongly believe that augmenting the acceleration data collection, CNN would play a key role for real time biofeedback on ISS. Using Convolutional Neural Networks, different outputs coming from several combination of sensors were tested. Discrete outcomes were reached even with sensors placed in homolateral position (in particular, related to Sacrum, Left Thigh, Left Shank and Sternum body points). However, the final goal regard the introduction of IMU sensors to be used with ARED on ISS. In this case, the homolateral solution is not recommended since the asymmetry would imply the exclusion of feedbacks related to one body side (left or right, depending on the sensors placement). Anyway, discrete results in terms of accuracy using CNN have been observed with IMUs on the lower limbs (84.29% for NS, 81.68% for WS and 76.92% for DL). Since customization is one of the MARS-PRE project's aims, the current thesis isolated three subjects (among the 17 available of the IMU acquisition)

who performed the highest number of repetitions and studied them separately. Hence, the ML algorithms previously mentioned were employed to build a customized classifier suited for a single person and good accuracy results were obtained (3 Female subjects, mean: 25.00 ± 3.61 years old, 58.00 ± 6.08 kg, 166.33 ± 3.21 cm height. Mean of accuracy for SVM and CNN, respectively: 98.61% ± 2.41% and 95.83% ± 4.17% for NS; 98.61% ± 2.41% and 87.5 % ± 4.17% for WS; 91.67% ± 3.61% and 87.5% ± 10.83% for DL). In addition, IMU and simulated data show visible differences: training ML algorithms with the former and testing with the latter (and viceversa) is not a reliable approach. Consequently, training and testing steps have been conducted only on the simulated accelerations (with discrete accuracy results for both SVM and CNN: 90.00% and 80.00% for Normal Squat; 85.71% and 71.43 % for Wide Squat; 75.00% and 75.00% for Deadlift), but the number of observations (almost 30 repetitions in total for each exercise) is too little, especially for the neural network. Unfortunately, the spread of Covid-19 virus imposed strict limitations thus excluding the possibility to increase the acquisitions for both IMU and simulated data.

Outlines of Thesis

The present work is structured as follow:

- Chapter 1: description of physiological adaptations that occur during space missions; review and state of art of countermeasures and motion capture systems used on ISS.
- Chapter 2: aim of the thesis.
- Chapter 3: description of the software used for biomechanical modeling and to conduct the simulations.
- Chapter 4: description of the set-up data collection and their processing; development and validation of a multi-label classifier.
- Chapter 5: results and discussions.
- Chapter 6: conclusions and future works.
- Appendix A.
- Appendix B.

SOMMARIO

Stato dell'arte e obiettivi

L'interesse per l'esplorazione spaziale relativa agli ultimi 60 anni ha dimostrato come l'umanità cerchi sempre di raggiungere l'impossibile. Nonostante lo studio di altri pianeti differenti dalla Terra sembrasse irrealizzabile, oggigiorno la possibilità di raggiungere Marte sta diventando una realtà sempre più concreta. Tuttavia, un progetto così ambizioso è imprescindibilmente legato a spiacevoli svantaggi, in termini di salute e adattamento ad un ambiente in microgravità. Tra tutti gli svantaggi nel vivere in un luogo privo di gravità, potremmo elencare: problemi cardiovascolari, respiratori, visivi e muscolo-scheletrici. Fra questi, ad esempio, molti effetti collaterali riguardano la Sindrome da Adattamento Spaziale, la ventilazione non uniforme, la perfusione irregolare, il decondizionamento muscolare, l'atrofia e la demineralizzazione ossea. Per scopi di ricerca, molti progetti hanno cercato di replicare e studiare le stesse condizioni ambientali trovate dai viaggiatori spaziali. Inizialmente, si pensava che l'immersione in acqua fosse un modello logico per ridurre l'attrazione di gravità sulla massa corporea, ma presto si è rivelata un metodo poco pratico e inefficiente. Il riposo forzato a letto, la cui longevità precede l'esplorazione spaziale, è diventato in seguito un'applicazione pratica per simulare le condizioni di assenza di gravità sulla Terra. Con il progresso dell'Informatica, anche le simulazioni via software (e.g. OpenSim) sono diventate un pilastro fondamentale in questo contesto.

Per prevenire il decondizionamento nelle future missioni di lunga durata (LDM), i membri dell'equipaggio seguono perciò uno specifico protocollo di addestramento pre-, durante e post-volo. In effetti, la NASA sta utilizzando la Stazione Spaziale

Internazionale (ISS) per ottenere più informazioni riguardo le contromisure fisiologiche da adottare. Sia l'allenamento cardiovascolare sia quello resistivo vengono eseguiti con specifici dispositivi: sono a disposizione un cicloergometro, CEVIS (Cycle Ergometer with Vibration Isolation and Stabilization), ed un treadmill, COLBERT, per il mantenimento della salute del sistema cardio vascolare; l'Advance Resistive Exercise Device (ARED) viene invece utilizzato per prevenire l'atrofia muscolare e la perdita di minerali ossei. ARED simula l'uso di pesi esterni generando un carico costante, che può variare da 0 a 272,5 Kg. Consente inoltre di eseguire fino a 29 esercizi tra cui Normal Stance Squat, Wide Stance Squat e Deadlift. Tale architettura si integra anche con un nuovissimo sistema di acquisizione del movimento (MOCAP), ovvero BTS SMART-DX. Quest'ultimo raccoglierà dati cinematici sulla base di sensori passivi posizionati sul corpo dei cosmonauti.

Questa tesi è inserita in entrambi i progetti di ricerca “MARcatori biologici e funzionali per la biomedicina aStronautica di PREcisione - MARS-PRE”, proposti dall'Agenzia Spaziale Italiana (ASI) e “ARED Kinematics – Biomechanical quantification of bone and muscle loading to improve the quality of microgravity countermeasure prescription for resistive exercise”, che coinvolge Agenzia Spaziale Europea (ESA), Neuroengineering and Medical Robotics Laboratory (NearLab) del Politecnico di Milano, Jhonson Space Center (JSC) of National Aeronautics and Space Administration (NASA) e Kayser Italia. Il primo progetto presenta come obbiettivo quello di trovare indicatori biologici e funzionali del danno muscolo-scheletrico, che può verificarsi durante una imprecisa esecuzione della routine di allenamento quotidiana sulla ISS. Quindi, un sistema in tempo reale interagirà e fornirà informazioni/avvisi sull'esecuzione delle prestazioni corrette/non corrette. Il secondo progetto, ARED-K, prevede di migliorare l'efficacia degli allenamenti quotidiani individuali sulla ISS in modo soggetto-specifico, stimando le forze interne in gioco. Per raggiungere questi scopi, è necessaria la raccolta di dati utilizzando un sistema di acquisizione del movimento e piattaforme di forza (pre-, durante e post-volo). Verranno poi condotte delle analisi biomeccaniche e dei confronti statistici tra i risultati ottenuti.

Attualmente, dati provenienti da sensori inerziali (collezionati durante gli esercizi eseguiti con ARED) non sono disponibili, ma in compenso vi sono quelli cinematici (ottenuti utilizzando il sistema MOCAP). Per questo motivo, è necessario sviluppare un modello biomeccanico per simulare il posizionamento dei sensori sul corpo ed estrarre dati analoghi.

Questo lavoro, perciò, si è quindi concentrato sui seguenti obiettivi: (1) miglioramento di un modello utile per condurre un'analisi biomeccanica del movimento umano in condizioni di microgravità sulla ISS; (2) simulazione di sensori inerziali in diversi punti del corpo, ottenendo perciò segnali di accelerazione relativi a corrette e scorrette esecuzioni di esercizi; (3) la realizzazione di un sistema basato sulla classificazione tramite algoritmi di Machine Learning che interagirà e fornirà informazioni/avvertimenti su performance corrette/non corrette durante la tipica routine di allenamento (e.g., quando si eseguono Normal Squat, Wide Squat e Deadlift), (4) Una volta scelto l'algoritmo di ML più adatto, validazione di un classificatore personalizzato e specifico per soggetto atto a riconoscere esecuzioni di esercizi corrette/non corrette.

Simulazione sensori inerziali, dati IMU e protocollo sperimentale

OpenSim è il software open-source utilizzato per condurre l'analisi biomeccanica e le simulazioni in microgravità. La piattaforma permette, grazie allo strumento “Analyze”, di simulare il posizionamento dei sensori su specifici punti del corpo. Quest’ultimi sono: sterno, sacro, coscia superiore destra, coscia superiore sinistra, stinco superiore destro e stinco superiore sinistro. In questo modo, è possibile estrarre i dati di accelerazione simulati. Il modello muscolo-scheletrico generalmente utilizzato per le attività di cammino e corsa è il "Gait2392", ma non è adatto ad esercizi quali lo squat profondo. In letteratura, il modello “Catelli” (già validato) ha dimostrato di aumentare i ROM articolari, soprattutto quelli relativi alla articolazione del ginocchio, permettendo così di sfruttarlo per i movimenti considerati in questo progetto di tesi. In particolare, questo modello è stato validato specificatamente per esercizi di Squat, i quali richiedono ROM più elevati per l'articolazione del

ginocchio, differentemente dal cammino e dalla corsa. Abbiamo anche deciso di verificarne la validazione, ottenendo esito positivo e conforme a quanto esposto nel paper di ricerca. Per raggiungere tale obbiettivo, sono stati considerati dati simulati provenienti da 6 soggetti reclutati al NASA JSC (3 Uomini, Media: 32.33 ± 3.06 età, 79.96 ± 10.84 kg, 174.67 ± 8,50 cm altezza; 3 Donne, Media: 31.33 ± 6.80 età, 60.20 ± 7.56 kg, 161.83 ± 7.21 cm altezza), i quali hanno eseguito 4 ripetizioni per ogni tipologia di movimento (Nromal Squat, Wide Squat e Deadlift) usando ARED.

Per quanto concerne le accelerazioni reali acquisite mediante sensori IMU, invece, abbiamo cominciato ad analizzare dati grezzi (raccolti a Pavia) di un altro progetto di tesi svolto in parallelo a questo ed inserito nella ricerca MARS-PRE. Dal lavoro di Ravizza et al. (2020), nessuna differenza statistica è stata osservata tra la cinematica degli esercizi eseguiti con ARED e quelli condotti con bilanciere. Di conseguenza, questo progetto di tesi si è basato su tale consapevolezza per condurre lo step relativo alle acquisizioni. Il sistema elettronico utilizzato (marca XSENS) per collezionare i dati di accelerazione è composto da 6 differenti accelerometri posti in differenti parti del corpo umano (1 su ogni coscia, 1 su ogni polpaccio, 1 sullo sterno. 1 sul sacro). Sono stati coinvolti solo 17 soggetti (9 Uomini, Media: 26.89 ± 5.73 età, 64.22 ± 7.14 kg, 173.44 ± 4.25 cm altezza; 8 Donne, Media: 25.38 ± 3.77 età, 56 ± 6.31 kg, 163.14 ± 6.52 cm altezza). Questi hanno eseguito un numero diverso di ripetizioni (il range varia da 5 a 20, in base alle loro capacità fisiche) di Normal Squat, Wide Squat e Deadlift (analogamente a quelli eseguiti al NASA JSC) usando bilanciere e pesi (con carico nel range 50-75% di ISO-MAX). I dati in questione non si riferiscono semplicemente alla corretta esecuzione degli esercizi, ma anche ad un dataset di allenamenti scorretti (evitando sempre di danneggiare in modo permanente o temporaneo le strutture fisiologiche dei soggetti reclutati), i quali sono stati approvati dagli specialisti della NASA JSC e dal Comitato Etico del Politecnico di Milano.

Elaborazione dati e sviluppo classificatore

Matlab R2019b è stato in seguito adoperato per pre-elaborare i dati IMU di accelerazione. I dati grezzi sono stati inizialmente filtrati con un filtro Butterworth

passa basso di sesto ordine, con una frequenza di taglio di 5 Hz. Dopo la rimozione di outlier e la considerazione di soli dati significativi, ogni singola ripetizione degli esercizi è stata isolata e ridimensionata a un valore di tempo medio. Questi passaggi consentono un utilizzo più rapido per la sezione successiva: Google Colab (basato su script python) è stata scelta come piattaforma candidata per normalizzare, applicare un algoritmo in grado di risolvere problemi di classificazione sbilanciati e, infine, per eseguire estrazione e selezione delle feature (PCA). Quest'ultimo step consente di ridurre il set delle feature ed è stato applicato a tutti gli algoritmi esplorati in questo studio (e.g., SVM, KNN, Decision Tree, XGBoost, MLP) ad eccezione della Convolutional Neural Network (CNN), che ha semplicemente utilizzato i dati pre-elaborati. Tutti gli approcci di apprendimento supervisionato sono stati poi sviluppati e testati per eseguire una classificazione multi-label, al fine di discriminare esercizi corretti e scorretti (rispettivamente 5 e 3 movimenti imprecisi sono stati classificati per Normal/Wide Squat e Deadlift).

Risultati e conclusioni

Modello biomeccanico idoneo agli esercizi dell'allenamento sulla ISS - Il modello biomeccanico OpenSim "Catelli", a differenza di "gait2392", si è dimostrato un valido candidato nel progetto di ricerca ARED-K in quanto permette di raggiungere quei Range di Movimento articolari (ROM), in particolare per la articolazione del ginocchio, adatti per gli esercizi specifici di allenamento (Normal Squat, Wide Squat, Deadlift) sulla ISS.

Classificazione delle prestazioni in base ai segnali di accelerazione - Il set di dati originale, contenente 1314 feature per sei sensori (quindi 219 feature per ognuno), viene ridotto con la PCA a 198, 216 e 152 feature, rispettivamente per Normal Squat, Wide Squat e Deadlift (descrivendo così per ognuno almeno il 90% della varianza dei dati). SVM e CNN hanno permesso, rispettivamente, di raggiungere risultati ammissibili e affidabili in termini di accuratezza: 87.79% e 84.30% per il Normal Squat; 86.38% e 86.91% per il Wide Squat; 83.76% e 82.05% per il Deadlift. Sono necessarie ulteriori indagini per ampliare il dataset e perfezionare il classificatore, ma

questi risultati preliminari potrebbero essere visti come un incentivo a considerare tale approccio una soluzione tangibile. Crediamo fermamente che ampliando la quantità di dati di accelerazione a disposizione, la CNN assumerà un ruolo chiave per un biofeedback in tempo reale sulla ISS. Utilizzando reti neurali convoluzionali, sono stati testati inoltre diversi output provenienti da svariate combinazioni di sensori. Risultati accettabili sono stati raggiunti anche con sensori posti in posizione omolaterale (in particolare, relativi ai punti del corpo dell'osso sacro, della coscia sinistra, dello stinco sinistro e dello sterno). Comunque, l'obbiettivo finale riguarda l'introduzione di un set-up composto da sensori inerziali IMU utilizzabili in combinazione con ARED. In tale scenario, la soluzione omolaterale è sconsigliata in quanto l'asimmetria comporterebbe l'esclusione dei feedback relativi ad un lato del corpo (sinistro o destro, a seconda del posizionamento dei sensori). Inoltre, usando la CNN, sono stati osservati risultati discreti in termini di accuratezza considerando sensori IMU posti sugli arti inferiori (84.29% per NS, 81.68% per WS e 76.92% per DL). Poiché la personalizzazione è uno degli obiettivi del progetto MARS-PRE, tre soggetti (fra i 17 disponibili) che hanno eseguito il maggior numero di ripetizioni sono stati isolati e studiati separatamente. Pertanto, gli algoritmi di ML precedentemente citati sono stati utilizzati per costruire un classificatore personalizzato adatto a una singola persona e buoni risultati di accuratezza sono stati ottenuti (3 Donne, Media: 25.00 ± 3.61 età, 58.00 ± 6.08 kg, 166.33 ± 3.21 cm altezza. Media di accuratezza per SVM e CNN, rispettivamente: 98.61% ± 2.41% e 95.83% ± 4.17% per NS; 98.61% ± 2.41% e 87.5% ± 4.17% per WS; 91.67% ± 3.61% e 87.5% ± 10.83% per DL). Inoltre, i dati IMU e quelli simulati mostrano differenze visibili: addestrare algoritmi di machine learning con il primo e testare con il secondo (e viceversa) non è un approccio affidabile. Di conseguenza, le fasi di training e test sono state condotte solo sulle accelerazioni simulate (con risultati di accuratezza discreti sia per SVM sia per CNN: 90.00% e 80.00% per il Normal Squat; 85.71% e 71.43 % per il Wide Squat; 75.00% e 75.00% per il Deadlift), ma il numero di osservazioni (in totale circa 30 ripetizioni per ogni tipologia di esercizio) è troppo basso, specialmente per la rete neurale. Sfortunatamente, la diffusione del

virus Covid-19 ha imposto rigide costrizioni escludendo così la possibilità di espandere il dataset sia relativo ai sensori inerziali sia a quello dei dati simulati.

Struttura della tesi

Il presente lavoro è strutturato come segue:

- Capitolo 1: descrizione degli adattamenti fisiologici che si verificano durante le missioni spaziali; revisione e stato dell'arte delle contromisure e dei sistemi di motion capture utilizzati sulla ISS.
- Capitolo 2: scopo della tesi.
- Capitolo 3: descrizione del software utilizzato per la modellazione biomeccanica e per condurre le simulazioni.
- Capitolo 4: descrizione del set-up adottato per la raccolta dei dati e loro successiva elaborazione; sviluppo e validazione di un classificatore multi-label.
- Capitolo 5: risultati e discussioni.
- Capitolo 6: conclusioni e sviluppi futuri.
- Appendice A.
- Appendice B.

LIST OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ANN	Artificial Neural Network
ARED	Advanced Resistive Exercise Devices
ASCR	Astronaut Strength, Conditioning and Rehabilitation
ASI	Agenzia Spaziale Italiana
ASIS	Anterior Superior Iliac Spine
BOS	Bar Over Shoulder
CEVIS	Cycle Ergometer with Vibration Isolation and Stabilization
CMC	Compute Muscle Control
CNN	Convolutional Neural Network
CO	Correct
DL (or ND)	Deadlift
ESA	European Space Agency
GRF	Ground Reaction Force
HAR	Human Activity Recognition
HB	Hyperextended Back
HDBR	Head-down bed rest
ID	Inverse Dynamics
IK	Inverse Kinematics
IMU	Inertial Measurement Unit
IR	Infrared
iRED	interim Resistive Exercise Device
ISO-MAX	Maximal Isometric Strength
ISS	International Space Station
JSC	Johnson Space Center
KNN	K-Nearest Neighbor
KOT	Knees Over Toes

KV	Valgus Knees
LDM	Long Duration Missions
LEAD	Lower Extremity Assistive Device
MARS-PRE	MARcatori biologici e funzionali per la biomedicina aStronautica di PREcisione
MLP	Multi-Layer Perceptron
MOCAP	MOtion CAPture
MSKM	Musculoskeletal Model
NASA	National Aeronautics and Space Administration
NS	Normal Squat
PCA	Principal Component Analysis
RB	Rounded Back
RH	Raised Heels
ROM	Range of Motion
RRA	Residual Reduction Algorithm
SIMM	Software for Interactive Musculoskeletal Modeling
SVM	Support Vector Machine
TVC	TeleVision Camera
VIS	Vibration Isolation System
WP	Word Package
WS	Wide Squat
XGBoost	Extreme Gradient Boosting

CHAPTER 1
STATE OF THE ART

In this first chapter we will expose the current state of the art essential to understand the roots that led to the budding of the current project.

1.1 Physiological adaptations during long-term space mission

During the last decades, many experiments and analysis have been conducted on the physiological modifications related to microgravity environments. These studies were possible thanks to mankind aim of exploring the space around our planet. Humanity firstly started with missions around the Earth Orbit: just think about Yuri Gagarin, who completed in 1961 a single orbit of the Earth for the first time. After this important accomplishment, the idea of reaching farther distances became concrete and real. Therefore, short-term and, later, long-term spaceflight (typically six months) became feasible, implying a wide variety of psychological and physical stressors on the body, including (but not limited to) sustained levels of ionizing radiation, circadian shifts, microgravity, diet restriction, sleep deprivation, reduced physical work, confinement and isolation. Space Motion Sickness is experienced by 60% to 80% of space travelers during their first 2 to 3 days in microgravity. It manifests clinically with symptoms similar to other forms of motion sickness (such

as malaise, fatigue, loss of appetite, nausea and vomiting) and it is a branch of the Space Adaptation Syndrome. The latter also includes facial stuffiness from headward shifts of fluids, headaches, and back pain (Heer et al. 2006). Among all the drawbacks and side effects while living in a place lacking gravity, we could list cardiovascular, respiratory, visual and musculo-skeletal downsides.

Cardiovascular alterations from spaceflight begin immediately upon liberation from Earth's gravitational force. Prolonged exposure to microgravity and radiation yields profound effects on the cardiovascular system, including a massive cephalad fluid translocation and altered arterial pressure, which attenuate blood pressure regulatory mechanisms and increase cardiac output. Also, central venous pressure decreases due to the loss of venous compression. The stimulation of baroreceptors by the cephalad shift results in an approximately 10%–15% reduction in plasma volume, with fluid translocating from the vascular lumen to the interstitium (Vernice et al., 2020).

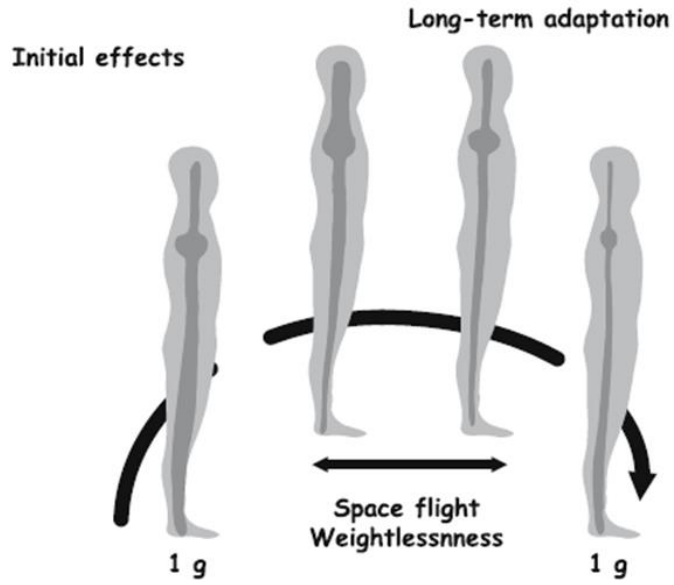


Figure 1: fluid shift from the lower to the upper part of the body in weightlessness environment. This thoraco-cephalic fluid shift stimulates central volume carotid, aortic and cardiac receptors inducing an increase in diuresis and natriuresis and a decrease in plasma volume (Narici et al. 2007). On Earth, the arterial pressure is higher in the feet and lower in the head, 200 mmHg and 70 mmHg respectively. In orbit, instead, the situation changes: the gradient of pressure is almost absent, hence the brain is subjected to a higher pressure with respect to the one present in normal conditions (Hargens *et al*., 2012).

As regards the ventilation apparatus, studies in microgravity condition have been performed and resulted in the so-called Slinky effect. The latter causes both uneven ventilation (through the deformation of lung tissue) and irregular perfusion (through a combination of the Slinky effect and the zone model of pulmonary perfusion). However, gravity serves to maintain a degree of matching of these two processes, in a way that the ventilation/perfusion ratio (and consequently the gas exchange) remains efficient. Therefore, gas exchange in spaceflight is quite similar to the one on Earth. Despite the changes in its function when gravity is removed, the lung continues to work properly in weightlessness without any apparent degradation even after 6 months in space (Prisk, 2014).

Table 1: lung function measurements performed in zero gravity (Prisk, 2019).

Topic	Key results
Lung volumes	Vital capacity unaltered after an initial reduction
	Functional residual capacity intermediate between standing and supine positions
	Residual volume reduced
Forced spirometry	Modest changes possibly consistent with increased lung water
Cardiac output	Increased initially with subsequent decline
Diffusing capacity	Sustained increase
	Increases in both components (membrane diffusing capacity and pulmonary capillary blood volume)
Heterogeneity of ventilation	Reduced in vital capacity breaths
	Minimal changes in tidal volume breaths
Ventilation heterogeneity in lung periphery	Altered in sustained but not transient zero gravity
Heterogeneity of perfusion	Reduced but not absent
Gas exchange and ventilation–perfusion matching	Largely unchanged in zero gravity
Control of ventilation	Reduction in hypoxic but not hypercapnic ventilatory responses
Sleep disordered breathing	Evidence for a reduction in zero gravity
Heart rate variability	Unaltered respiratory sinus arrhythmia in-flight, reduced post-flight
Chest wall mechanics	Increase in abdominal contribution to breathing
Extravehicular activity	No pulmonary disruption
Long duration zero gravity	Minimal sustained changes post-flight

Considering the visual system, research projects lead by NASA compared two twin subjects spending time on ISS and on Earth, respectively. Increases in subfoveal choroidal thickness (the primary vascular supply of the outer retina) and peripapillary

total retinal thickness were observed (referring to the space traveler), thus indicating retinal edema formation. Moreover, the severity of the choroidal folds augmented during spaceflight. These ocular structure parameters were unchanged on ground for the whole duration of the study. Of relevance to these ocular alterations, untargeted proteomics revealed a decrease in urine leucine-rich alpha-2-glycoprotein (LRG1) levels inflight. LRG1 itself has been reported to play a role in retinal vascular pathology (X. Wang et al. 2013). The predisposition to develop these ocular changes during spaceflight has also been associated with B-vitamin status and the presence of specific single nucleotide polymorphisms (SNPs) (Zwart et al, 2016). Risk alleles in five SNPs predict incidence of ophthalmic issues (X. Wang et al. 2013), and six of the nine risk alleles are present in the twins (Zwart et al, 2016). Furthermore, serum folate, a B-vitamin that is often low in astronauts who experienced ophthalmic changes during flight (Zwart et al. 2012), was also low in both twins.

Nowadays, a primary objective of the international space programs is to install permanently manned bases on the Moon and to undertake manned explorations of Mars within the next 5-10 years. For these projects to become a reality, the negative effects of microgravity on the human muscle system must be overcome. Indeed, since the beginning of the space-flight era, weightlessness was shown to lead to substantial changes of muscle function. These changes, globally defined as deconditioning, consist mainly of loss of muscle mass, force and power, increased muscle fatigability, and abnormal reflex patterns (for reviews see Ferretti et al., 1997). They are due to a combination of factors among which an increased degradation of muscle proteins and substantial changes of the neuromuscular control of movement (both brought about by the absence of the constant pull of gravity) play a major role. Uninterrupted bed rest (see chapter 1.2 for further information on this topic) was found to be a suitable model to simulate the decrease in proteins synthesis. The researchers showed a 50% reduction in skeletal muscle protein synthesis during the first two weeks of bed rest (Ferrando *et al.*, 1996). Hence, during long-term space missions, muscle deconditioning could limit the crews' ability to work in space (or on the Moon/Mars surface) and to rapidly egress the spacecraft in an emergency

landing. Furthermore, muscle atrophy and weakness are of particular concern when the transition from zero to one g occurs, as the musculoskeletal system after several days/months in weightlessness has to bear suddenly the gravity force.

There exists very limited data for determining the effectiveness of human health and performance countermeasures intended to preserve astronaut health during long duration space exploration missions. Exercise countermeasures used in the Space Shuttle Program and on the International Space Station do not eliminate bone loss or muscle deconditioning. Without an effective countermeasure, astronauts lose bone density at a rate of 1-2% a month, which may lead to early onset osteoporosis and place the astronaut at greater risk of fracture after returning to the earth's gravitational field.

1.2 Space environment simulation on ground

With the advent of human spaceflight in 1961, thousands of research projects started to recreate the same specific environmental conditions found by the space travelers. In this way, trials could be performed also on Earth, thus augmenting the number of experiments feasible (avoiding the exclusivity in-flight). Initially, immersion in water was used as a logical model for reducing the pull of gravity on the mass of the body. However, it soon resulted impractical remaining in water for more than a day since it brought on unpleasant consequences.

Figure 2: astronaut is preparing to train underwater. Credits: NASA.

Following, a couple different approaches are listed: the first one refers to a practical and concrete replication, while the second relies on a simulation performed via software (Matlab and OpenSim).

Bed rest

The history of bed rest predates spaceflight. In the nineteenth century, it was first introduced as a medical treatment, but then many noticed how resting was not the cure to every disease.

In the early 1970s, cosmonauts coming back from longer missions complained to their medical staff about a hard time sleeping since they felt the sensation of slipping off the foot of the bed. The latter was raised till the astronauts perceived a horizontal sensation: in this way they could go back to sleep. Every night they lowered the foot of the bed a little, in order to restore their habitual preflight behaviors. Russian researchers took note of this observation and theorized that, perhaps, the head-down

position on Earth was closer to what the crew felt in space. The HDBR simulation model was born.

Nowadays, bed rest studies allow scientists to analyze the adaptation of human body in weightlessness. HDBR is characterized by immobilization, inactivity, confinement and elimination of Gz (vertical axis) gravitational stimuli, such as posture change and direction (which affect body sensors and responses). These induce upward fluid shift, unloading the body's upright weight, absence of work against gravity, reduced energy requirements and reduction in overall sensory stimulation. The upward fluid shift (by acting on central volume receptors) induces a 10–15% reduction in plasma volume. This leads to a now well-documented set of cardiovascular changes including modifications in cardiac performance and baroreflex sensitivity (which are identical to those in space). Moreover, calcium excretion is increased from the beginning of bed rest leading to a sustained negative calcium balance. Contemporarily, calcium absorption is reduced. In addition, body weight, muscle mass and muscle strength are reduced, as the resistance of muscle to insulin. Bone density, stiffness of bones of the lower limbs, spinal cord and bone architecture are altered. Also, circadian rhythms are dampened and may shift (Narici et al. 2007).

Volunteers spend up to 70 days in bed with a six-degree head-down tilt. They have limited freedom of movement since they must eat, exercise, and even shower in the head-down position. In this way, their bodies adapt replicating almost the same environmental space condition. Continuously monitoring is fundamental to understand any possible body change together with the associated reason.

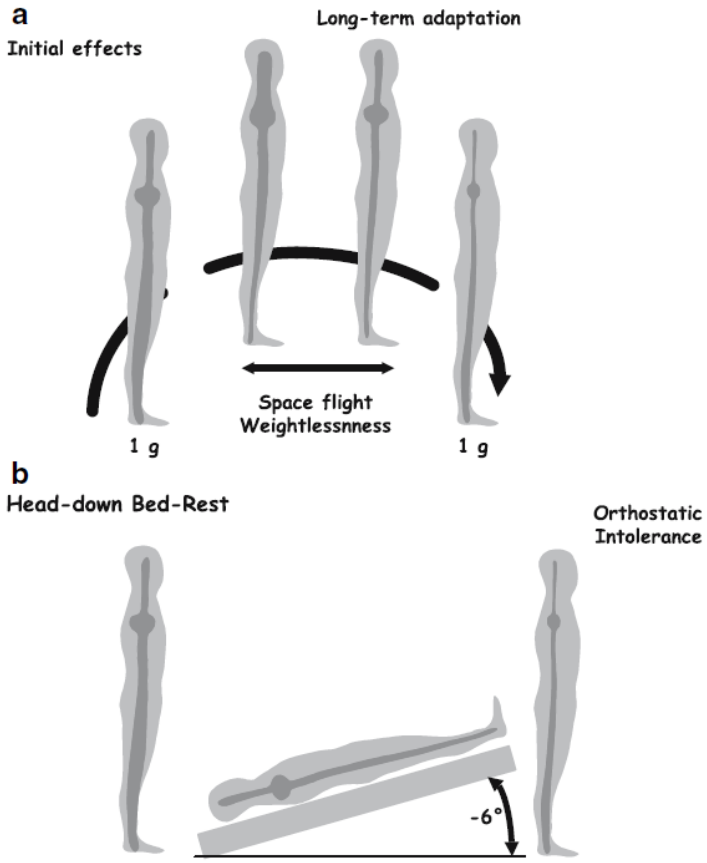


Figure 3: fluid shift from the lower to the upper part of the body induced by real (a) and simulated weightlessness (b). As in spaceflight, cardiovascular deconditioning characterized by orthostatic intolerance is observed at the end of bed rest (Narici et al. 2007).

Simulation of a LEAD for resistance training in microgravity environment via software

During a generic human body motion, the gravitational acceleration prevents muscle atrophy by continuously generating a resistance of the movement, granting the possibility to maintain contact with the ground during walking (thereby producing a ground reaction force, GRF). However, during long time periods of space flights, many issues arise: among them, muscles atrophy itself must be taken into account. Indeed, the resistance force is barely generated when moving in a microgravity environment. In order to prevent this phenomenon, Lower Extremity Assistive Devices (LEADs) can be used to provide a specific resistance training.

While developing an assistive device for physical workout in weightlessness, it is necessary to select the types and locations of appropriate actuators and determine the control inputs to implement the desired task. In the design process, musculoskeletal simulations can be used to examine the actuator locations and acquire the corresponding control input, which enables the tracking of the desired motion data and muscle forces.

The purpose of the simulation is allowing the human model to track the state variables (generalized coordinate, generalized speed, muscle activation and muscle fiber length) measured and predicted during 1 g normal motion with the assistance of an external actuator under microgravity conditions. Simulation for a LEAD in microgravity was performed in the research lead by Jong In Han (et al., 2020) with two platforms: Matlab and OpenSim (that allows to set at zero-g value the gravitational acceleration).

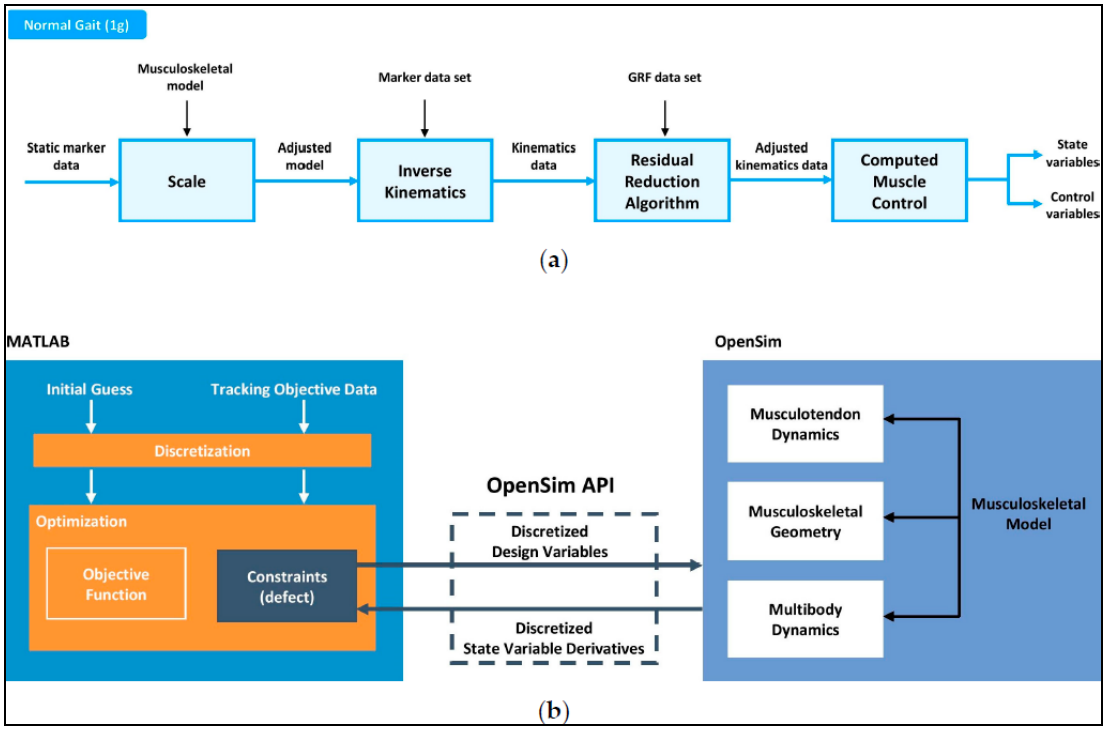


Figure 4: image taken from the research paper. It shows the processes used in OpenSim and Matlab (a) Process for obtaining tracking objective data. (b) Optimization process using Matlab and OpenSim (Jong In Han, 2020).

Having in mind all the information explained so far, it is evident the importance of specific training plans pre-flight, in-flight and post-flight. Pre-flight exercises have multiple purposes:

- support astronauts in maintaining an over-all fitness level.
- collect data to tailor individually the in-flight training protocols.
- prepare them by replicating the ISS exercise countermeasures.

Post-flight exercise reconditioning is necessary to correct any alteration resulting from the microgravity adaptation and the re-adaptation to Earth's gravity. However, in the current work we are going to briefly overview only the inflight phase.

1.3 In-flight countermeasures on ISS

The International Space Station (ISS) (figure 5) is a large spacecraft in orbit around Earth at an average altitude of approximately 250 miles and travels at 17,500 mph. The first piece of ISS was launched in November 1998.

Figure 5: a photograph of the International Space Station (ISS). Credits: NASA.

Nowadays, it serves as a home where crews of astronauts and cosmonauts live, and it is used also as a unique science laboratory (due to the peculiarity of being in absence of gravity). Indeed, the areas of research cover different fields: biology, life science, development and validation of new technologies, astronomy and Earth observation. Several nations worked together to build and use the space station. The latter is made up of parts assembled directly in space by the astronauts. In particular, it consists of 3 nodes: node 1 (Unity), that connects the U.S.A and Russian segments of the ISS; node 2, which connects the U.S.A, European and Japanese laboratories; node 3 (Tranquility), whose aim is about providing habitation functions including hygiene, sleeping and training compartments. The last one is actually the most modern of ISS: it was built by the European Space Agency (ESA) and the Italian Space Agency (ASI) and it hosts the training area of the Station (figure 7).
Hence, NASA is using the space station to learn more about living and working in a space environment. These lessons will make it possible to send humans at a farther distance from earth than ever before.

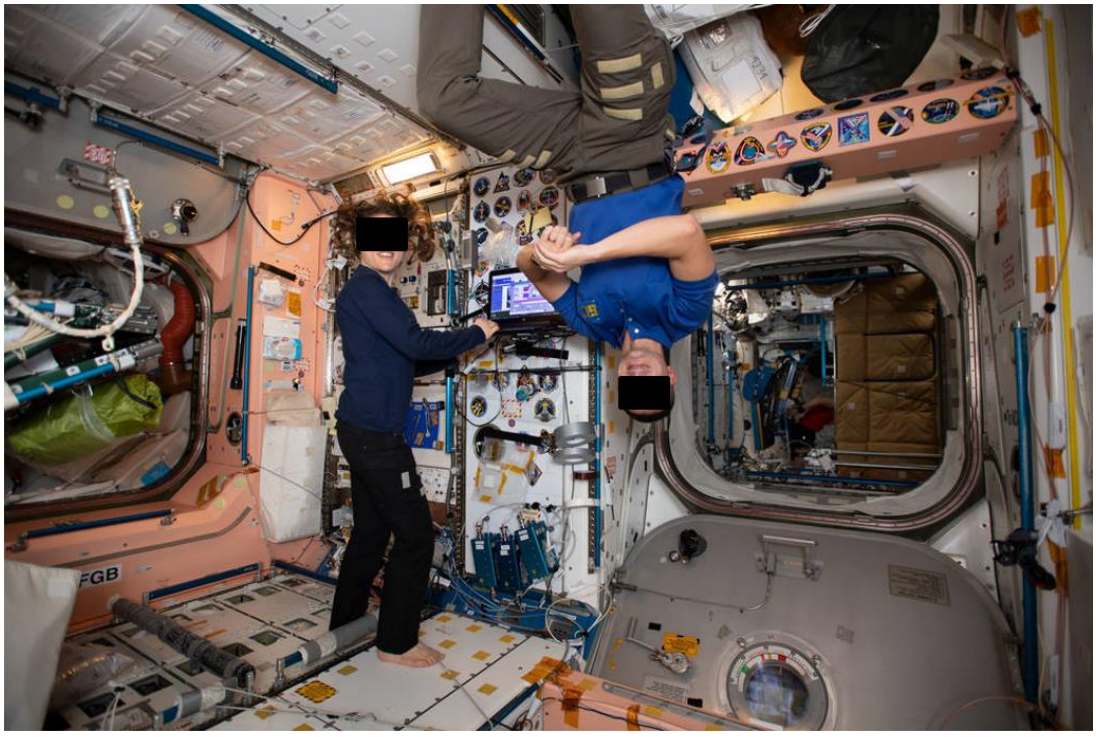


Figure 6: at the end of October 2020, 240 people from 19 countries have been on the International Space Station. More than 2,800 experiments have been conducted in space. Credits: NASA.

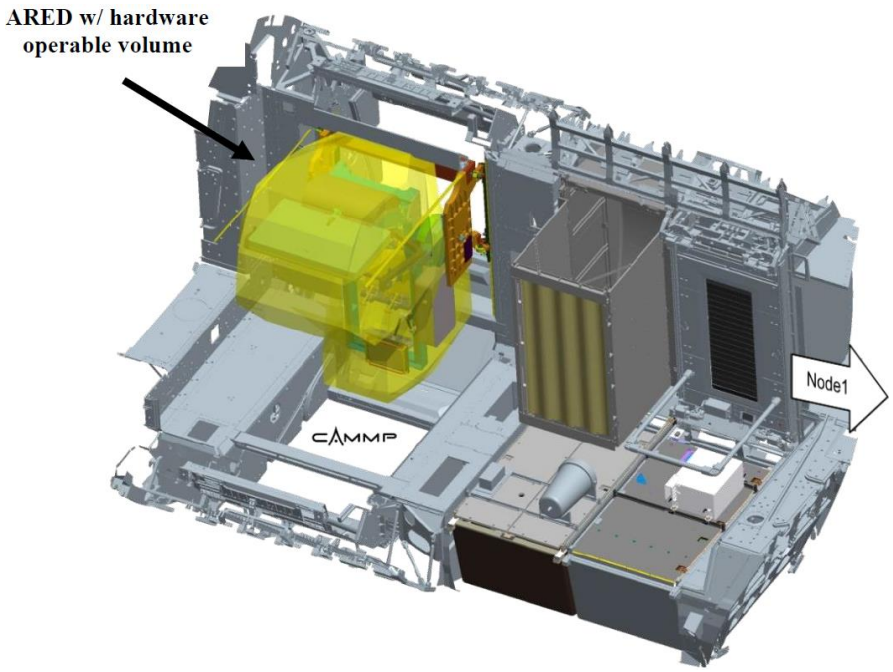


Figure 7: node 3 configuration with the Advanced Resistive Exercises Devices (ARED) system.

Innovation and progress allowed in the first two decades of the 21st century to improve the In-flight countermeasures exercise devices (as shown in table 2).

Table 2: historical overview of exercise countermeasure hardware available on ISS for ESA's eight long-duration missions to the International Space Station (ISS) (from Petersen *et al*., 2016). iRED: interim resistive exercise device; ARED: advanced resistive exercise device; TVIS: treadmill with vibration isolation and stabilization system; T2: 2nd generation treadmill; CEVIS: cycle ergometer with vibration isolation and stabilization system; VELO: Russian cycle ergometer.

Year	Hardware used by ESA crew on ISS	ESA mission
2000–2009	Treadmill (TVIS, BD-1)	LDM 1–3
2000–2009	Resistive exercise device (iRED)	LDM 1–3
2009–	Treadmill (T2), resistive exercise device (ARED)	LDM 3–8
2013–	Treadmill BD-2	LDM 6–8
2001–	Cycle ergometer (CEVIS, VELO)	LDM 1–8

Among the devices, we could count:

- Two cycle ergometers and two treadmills are available for cardiovascular exercises. The Cycle Ergometer with Vibration Isolation and Stabilization (CEVIS) and the VELOergometer provide workloads from 25-350 Watts and 100-250 Watts, respectively. The 2nd generation of treadmill, namely COLBERT or T2, and the D-2 treadmill provide motorized speed up to 20.4 km/h and 20 km/h, respectively.
- Resistance exercise was previously performed by using the interim Resistive Exercise Device (iRED), which was later substituted by the Advanced Resistive Exercise Device (ARED) (figure 8) in 2009 because of its inefficacy on maintaining muscle mass, muscle strength and bone mineral density (Lang *et al.*, 2004; Trappe *et al.*, 2009).

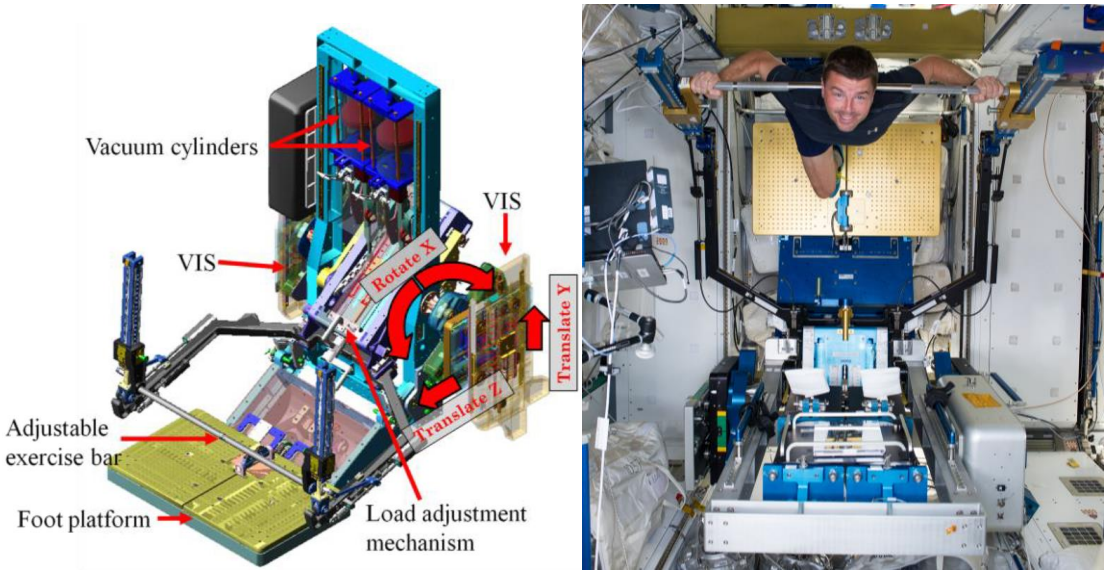


Figure 8: on the left, ARED/VIS module (Image Credit: NASA). The foot platform provides a support for the astronauts to perform their physical exercises and it is assembled with two force plates used to measure the Ground Reaction Forces (GRFs). Vacuum cylinders embedded in the system provide the force to the adjustable exercise bar, which includes the wishbone arm and the lift bar components. The load adjustment mechanism allows to simply modify the loads according to the needs of the space travelers. Vibration Isolation System (VIS) avoids any transmission of vibrations to the ISS by means of absorbing shocks. On the right, an astronaut is performing his own exercises using ARED (Image Credit: NASA).

ARED simulates the use of free weights in microgravity by generating a constant load. The resistive force is supplied by two piston-driven vacuum cylinders with an adjustable load: from 0 to, approximately, 2670 N (0 to 272.5 Kg on Earth) for bar exercises. In order to avoid any transmission of forces to the ISS during exercising, the ARED is attached to the station with a Vibration Isolation System (VIS), which absorbs shocks thanks to springs and dampers.
Moreover, ARED allows to carry out up to 29 exercises including Normal Stance Squat, Wide Stance Squat, Single Leg Squat and Deadlift. These exercises are performed by adding an external load as a percentage of individual maximal isometric strength (ISO-MAX) tested before flight.

Currently, the training approach is planned ad-hoc specifically for each astronaut and includes mainly three phases (Petersen *et al*., 2016) with a gradual load increment:

I. Adaptation phase: starting from the second day (after arrival on ISS), 2-3 weeks of cycle ergometer exercises are necessary. Initially, the astronaut is involved 1h per day, but later the training raises up to 2.5h per day. The intensity is relatively low.
II. Main phase: for 130 – 250 days, resistance exercise load has to be increased. Hence, every week the load itself is incremented by 3-5% of ISO-MAX, eventually reaching (at least) its 80%. Running and cycling speeds are enhanced always basing on the individual performance.
III. Preparation for re-entry phase: during the last 3-4 weeks of flight the loads of resistive exercises are maintained high.

A critical drawback to be considered in a microgravity environment regards the body weight: the latter is not perceived as on Earth. The reason is related to a reduced (or absent) gravity acceleration. Therefore, the application of loads via the ARED bar becomes fundamental. At the moment, a replacement of almost 70-75% of body weight is used, basing on the personal experience and feedback of astronaut.

Target exercises on ISS

Normal Stance Squat, Wide Stance Squat and Deadlift can be considered actually the exercises generally performed on ISS during each astronaut's training session. As previously mentioned, the ARED system allows the space travelers to undergo extensive physical preparation to guarantee correct performance, thus avoiding any injury or inefficacy of their workout. In order for the readers to have a better understanding of the whole process, a practical explanation regarding the optimal techniques of those exercises is reported below (Ravizza et al., 2020). Keep in mind that the exercises themselves and their execution is reviewed and approved by the NASA Astronaut Strength, Conditioning, and Rehabilitation (ASCR) specialists.

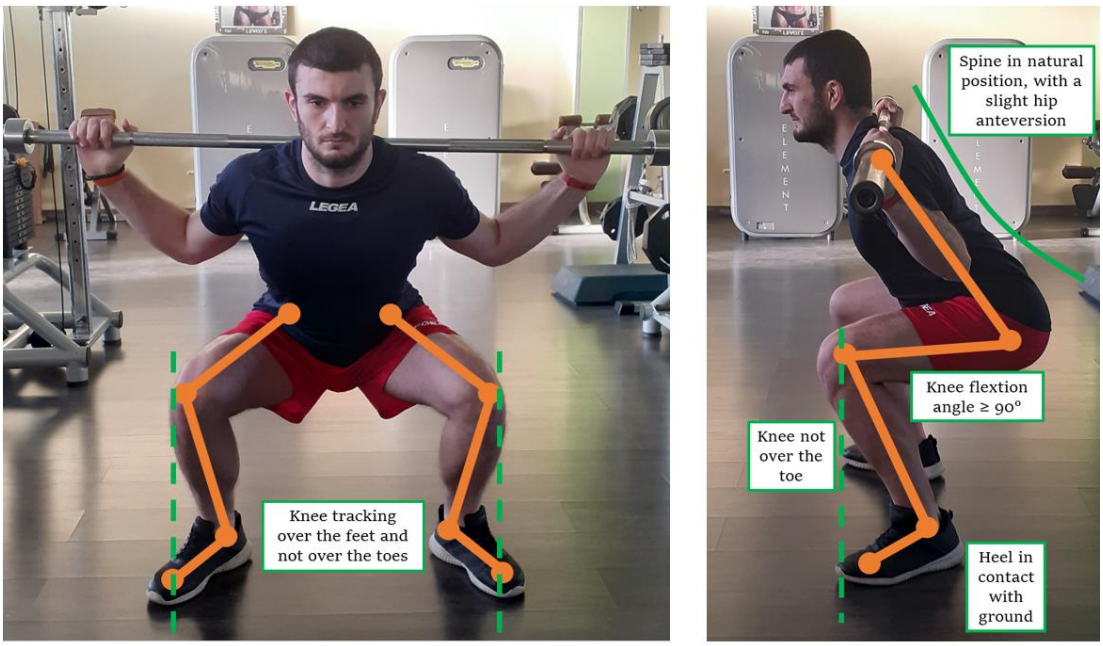
Normal stance squat and wide stance squat

The Normal Stance Squat, one of the main exercises used both in training and rehabilitation, guarantee to strengthen lower limb muscles and to assess ranges of motion. It is a closed kinetic chain motion task that involves hip and knee flexion together with ankle dorsiflexion. This movement implies both recruitment and synergy of several muscle groups, whose contraction and relaxation happens in specific motion phases. Beginning from a standing position and maintaining a straight trunk, the knees are flexed to reach an angle equal (or greater) than 90° and, after their extension, they reach the starting position again.

As far as Wide Stance Squat is concerned, instead, it can be considered a variation of the Normal Stance Squat already explained. It involves, indeed, more muscles than the first one. While in the Normal Stance Squat the feet placement is closer (a separation similar to the shoulders' one), the Wide Stance Squat requires more distance (a separation 1.5-2 times larger). Anyway, if both cases are performed with an inappropriate technique, they can provoke training inefficacy and/or several injuries (especially at levels of trunk and lower limb joints).

To minimize the risks of trauma and ensure maximal lower limb muscles activation, an optimal squat technique (figure 9) requires: heels in contact with the floor (to prevent forward lean of the trunk); upright trunk to maintain spine in a neutral

position, with a slightly lordotic lumbar spine (pelvic anteversion); knees tracking over the toes (preventing to bring them closer and avoiding to overcome the vertical lines of the toes themselves); tibiae parallel to the upright torso; gaze forwards or upwards (Comfort *et al*, 2018; Braidot *et al.*, 2007; Lorenzetti *et al.*, 2018; Myer *et al.*, 2014).



(a) (b)

Figure 9: frontal (a) and lateral (b) views of correct squat position. Dashed green vertical lines originate from the toes. In frontal plane, knee joints remain laterally with respect to the lines; in sagittal plane, they stay behind the lines. The solid green line highlights the natural lordotic lumbar curve, which has to be maintained during squatting. Knee angle should be ≥ 90°.

Deadlift

Deadlift is a training exercises commonly known and used to develop strength of trunk muscles. High risk of back injuries is possible when performing this complex motion incorrectly. The optimal deadlift technique (figure 10) requires an initial starting position in partial squatting, with natural width of feet and arms coming down outside the legs to reach the bar. Then, hip and shoulders have to be lifted at the same time maintaining a natural position of the spine.

Similarly to the squat exercise, also in this case it is necessary to respect some rules to avoid inefficacy of training and/or injuries: hip joints must be higher than knees, thus preventing forward lean of the trunk; upright trunk has to maintain spine in a neutral position; shoulder blades have to be adduced slightly in front of bar; one have to gaze forwards; knees must be tracking over the toes (preventing to bring them closer and avoiding to overcome the vertical lines of the toes themselves).

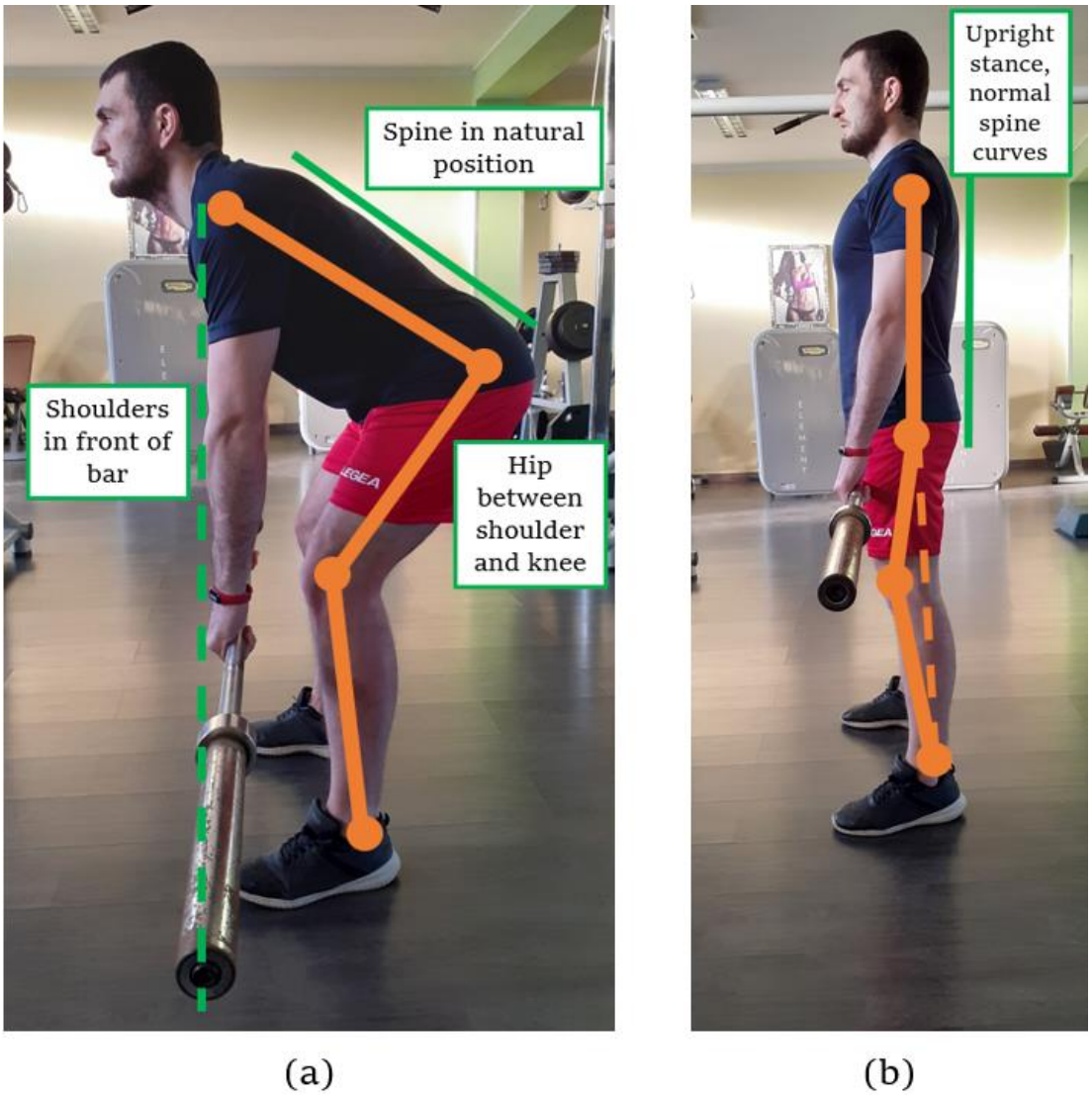


Figure 10: lateral view of correct starting (a) and ending (b) positions of Deadlift. Dashed green vertical line originates from the shoulder: the bar must not pass that boundary line. Solid green line shows the correct upright position of trunk to be maintained during lifting. Dashed orange line, instead, points out the correct alignment of shoulder, hip and knee at the end of exercise.

1.4 The importance of having a real-time feedback

Currently, inside the ISS, the ARED Photo/TV system (figure 11) is used in order to get a real time audio/video communication between the ground team and the astronauts during the training session. As we have already anticipated, feedbacks and recommendations are the main keys to maintain physical integrity. However, NASA and ESA are planning to travel for farther distances in terms of space missions: indeed, Mars is a target planet that humanity wants to reach and study. Of course, to succeed in the mission, a long journey will be necessary and the distance from Earth would increase significantly. Therefore, longer communication delays are expected, thus implying the inefficiency (due to the time delay itself) of those recommendation just mentioned above (Linh Vu *et al*., 2018). Consequently, it would be fundamental a specific tool able to provide real-time instructions and correction feedbacks for the space travelers without considering the human coaching anymore. In this way, we maintain intact the important aims about preventing possible injuries and optimizing the overall muscle strength outcomes.

Figure 11: standard ARED Photo/TV system field of view on ISS (from Ferchette *et al.*, 2017).

To achieve this purpose, on the ISS a computerized system able to detect and analyze each exercise repetition could implemented to grant a real-time corrective biofeedback. In the current thesis project, we tried to compare different Machine Learning tools combined with the data extracted from OpenSim (see chapter 3 and 4 for further information). Hence, the following paragraphs will describe the state of the art regarding different ML approaches for both offline and real time signals classification. As we are about to observe, those algorithms are used in many different applications and for various goals.

A first example can be seen in the 2020 article '' Automatic Classification of Squat Posture Using Inertial Sensors: Deep Learning Approach'' by Jaehyun Lee et. al, the squat posture classification performance of deep learning was compared to that of conventional machine learning. Accelerometer and gyroscope data were collected from 39 healthy participants using five inertial measurement units (IMUs) attached to the left thigh, right thigh, left shank, right shank, and lumbar region. Each participant performed six repetitions of an acceptable squat and five incorrect forms of squats that are typically observed in inexperienced exercisers. The accuracies of squat posture classification obtained using conventional machine learning and deep learning were compared.

Table 3: table taken from the Jaehyun Lee's research. It shows the squat classification performance of conventional machine learning (CML) and deep learning (DL) when considering five IMUs, two IMUs and one IMU.

Number of IMUs	Placement of IMUs	Random Forest (CML)			CNN–LSTM (DL)		
		Accuracy	Sensitivity	Specificity	Accuracy	Sensitivity	Specificity
5 IMUs	Right thigh, right calf, left thigh, left calf, and lumbar region	75.4%	78.6%	90.3%	91.7%	90.9%	94.6%
2 IMUs	Right thigh and lumbar region	63.2%	64.6%	87.6%	83.9%	85.6%	90.4%
	Right thigh and right calf	73.9%	76.8%	89.5%	88.7%	90.5%	95.7%
	Right calf and lumbar region	66.0%	70.1%	86.1%	86.2%	87.1%	87.6%
1 IMUs	Right thigh	58.7%	66.7%	88.9%	80.9%	80.0%	93.1%
	Right calf	57.6%	62.7%	82.2%	76.1%	78.9%	92.8%
	Lumbar region	34.6%	38.6%	68.1%	46.1%	50.3%	79.0%

Each result was obtained using one IMU or a combination of two or five IMUs. Overall, the deep learning approaches demonstrated to be superior with respect to outcomes obtained using conventional machine learning (for both single and multiple IMUs).

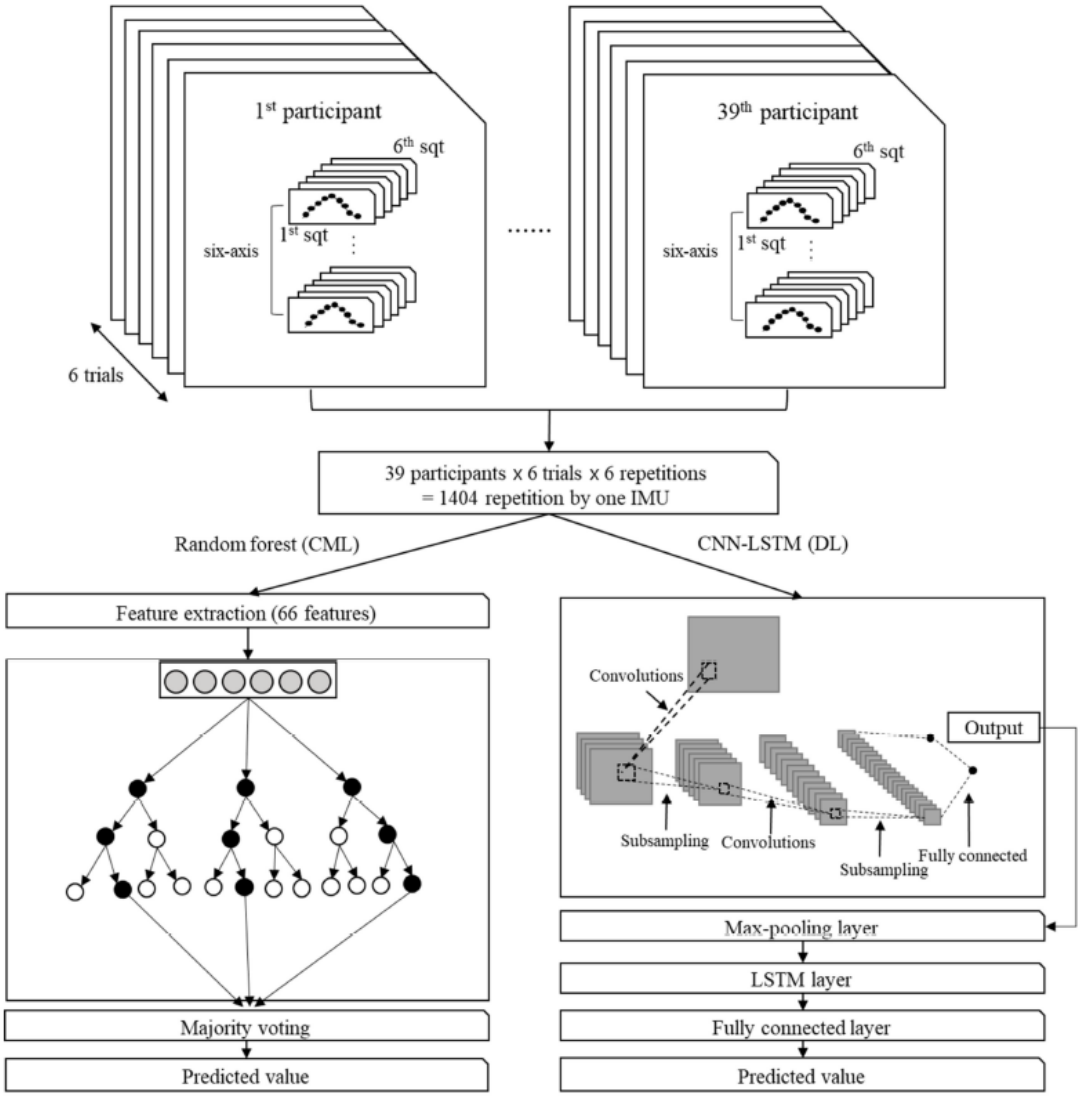


Figure 12: processes used to train classification models (random forest and convolutional neural network–long short-term memory (CNN–LSTM)) via segmented repetitions of squats.

A second example is shown in the research of Ashish Shrestha and Ji Dang on a 'Deep Learning-Based Real-Time Auto Classification of Smartphone Measured Bridge Vibration Data'.

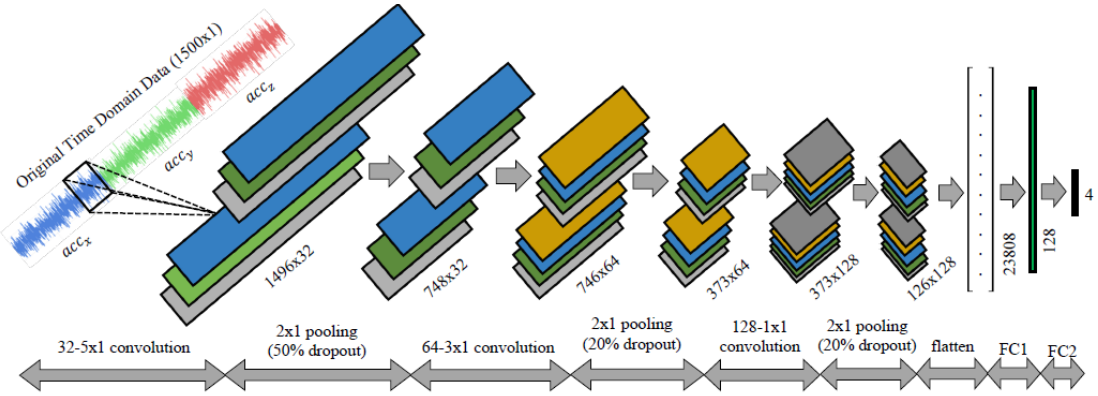


Figure 13: detailed map of convolution neural network (CNN) architecture for time series vibration classification.

In this study, a simple and customizable convolution neural network framework (for further information on CNNs see chapter 4) was used to train a vibration classification model that can be integrated into the measurement application in order to realize accurate and real-time bridge vibration status on mobile platforms. The inputs for the network model are basically the multichannel time-series signals acquired from the built-in accelerometer sensor of smartphones, while the outputs are the predefined vibration categories.

Table 4: table taken from the Ashish Shrestha and Ji Dang's research. It shows specifically how the network is composed.

Layer (Type)	Shape	Parameter
Conv1D (32 * 5 * 1)	1496 * 1 * 32	192
Max Pooling	748 * 1 * 32	0
Dropout (50%)	748 * 1 * 32	0
Conv1D (64 * 3 * 1)	746 * 1 * 64	6208
Max Pooling	373 * 1 * 64	0
Dropout (20%)	373 * 1 * 64	0
Conv1D (128 * 1 * 1)	373 * 1 * 128	8320
Max Pooling	186 * 1 * 128	0
Dropout (20%)	186 * 1 * 128	0
Flatten	23,808	0
Fully Connected 1	128	3,047,552
Fully Connected 2	4	516
Total Parameters: 3,062,788		

Both examples afore explained used acceleration data with gravity and did not consider external loads, differently from the current thesis project.

Motion Capture (Mocap) System on ISS

Motion Capture (also named as MO-CAP or MOCAP) is the process of digitally record the movement of people. It is used in entertainment, sports, medical applications, ergonomics and robotics. In film-making and game development, instead, it refers to recording actions of actors for animations or visual effects. A famous example of a movie with lots of motion capture technology is “Avengers: end game” (figure 14).

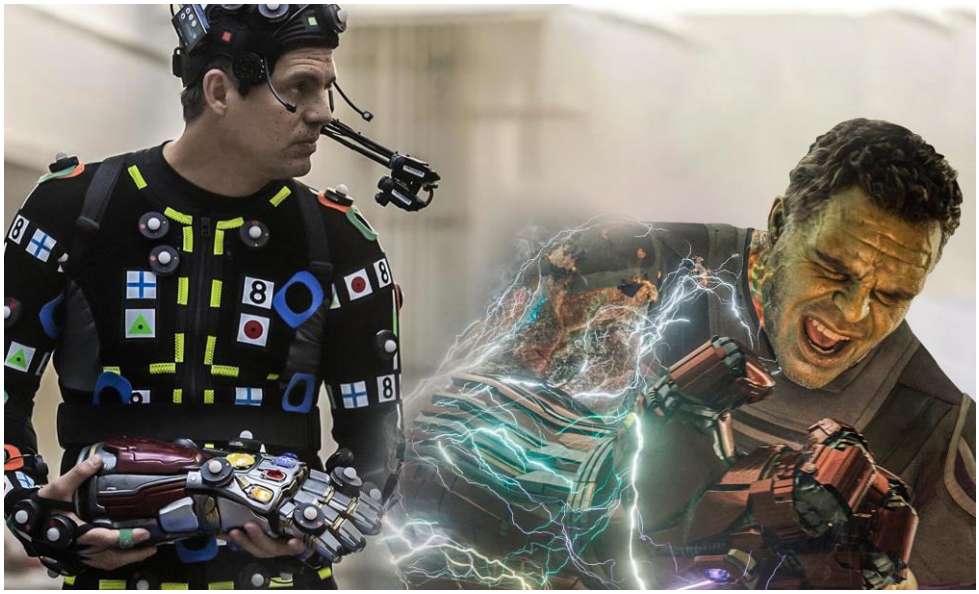


Figure 14: example of motion capture system used in filmography.

As regards the engineering and biomechanical world, this method permits to acquire kinematics data: joint positions, velocities, accelerations and angles. The analysis of those parameters allows to obtain biomechanical information about the musculoskeletal system during the execution of a motor task.

Three different branches of human motion tracking technologies (used for the whole body or just for a specific part of it) can be listed as follows:

- Marker-based systems.
- Depth camera-based systems.
- Sensor-based systems.

Marker-based motion capture systems have been used as the main tool in capturing motion for years. They are very pricey, lab-based and beyond reach of many researchers, hence they cannot be applied to ubiquitous applications. This kind of device exploit optoelectronic measurement systems (OMSs), markers located on the subject's body surface and a set of television cameras (TVCs). Every marker must be "seen" by at least one pair of TVCs: this allows to recognize in a three-dimensional way the markers' position in space (in terms of x y z, with respect to the reference axis or the laboratory axis).

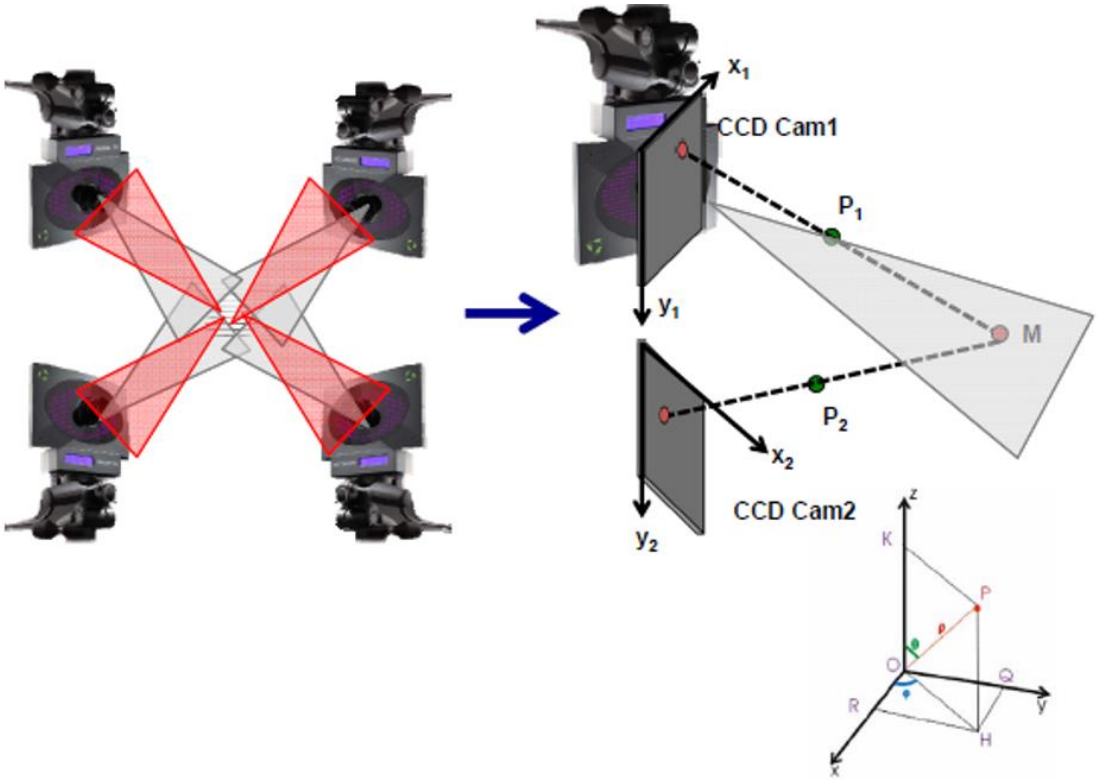


Figure 15: TVCs cameras allow to recognize in a three-dimensional way the markers' position in space.

These markers are actually little spheres which can be passive (hence coated with reflective substances) or active (able to emit lights by themselves).

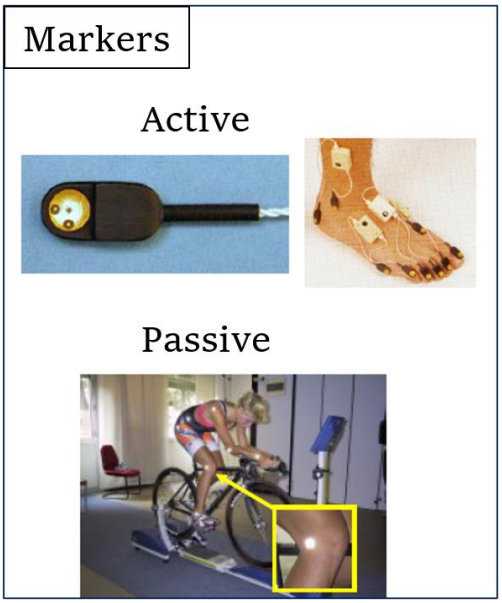


Figure 16: example of active and passive markers.

In order to have a better understanding of their differences, advantages and disadvantages are described below.

Advantages passive markers:

- Freedom of movements: neither cables nor batteries are needed.
- Absence of heat.
- "Unlimited" number of markers feasible since they do not need a power supply.
- Their functioning is independent from the number placed on the subjects.
- It is feasible to carry out evaluations both in total and specific body parts.

Disadvantages passive markers:

- Need of coaxial illuminators.
- Non-automatic labeling.
- Higher costs with respect to the active ones.
- Non-usable in open spaces: different noise sources must be considered such as sunlight, which can alter data acquisition.
- Proper calibration is needed to optimize and increase the accuracy.

Advantages of active markers:

- Automatic labeling: markers light up according to the position on the body.
- Real time tracking: immediate identification and representation on the laptop.
- Noise reduction: light is not a reflection that can be easily affected by noise.

Disadvantages active markers:

- Encumbrance and limitation of movements.
- Positioning time: the need also a power supply to be placed.
- Heat.
- Synchronization between markers since a certain lighting sequence is needed for their tracking.
- Adequate sampling frequency.

Examples of OMSs (used to perform kinematic analysis and scientific experiments with astronauts, on Earth and/or on ISS) are ELITE-S2, CODAmotion and BTS-SMART (figure 17).

Figure 17: TVC of BTS-SMART system.

Later, depth camera technologies introduced a new type of devices: we are talking about a motion capture system which is marker-less (yet cheaper), more portable and

quicker to setup. Among all the depth cameras emerged over the last years, the most notably are Structured Light (SL)-based and Time-Of-Flight (ToF)-based cameras. Indeed, recent Kinect v2, based on ToF technology, embeds an optical apparatus consisting of an RGB camera and a double infrared (IR) depth sensor. The latter is composed of an infrared projector and a camera sensitive to the same band, which in practice reads what is detected by infrared rays.

Kinect deduces the position of the body through relying on the following two steps:

1) A depth map is built using the structured light analysis created by the infrared emitter.
2) The position is inferred using specific tracking algorithms.

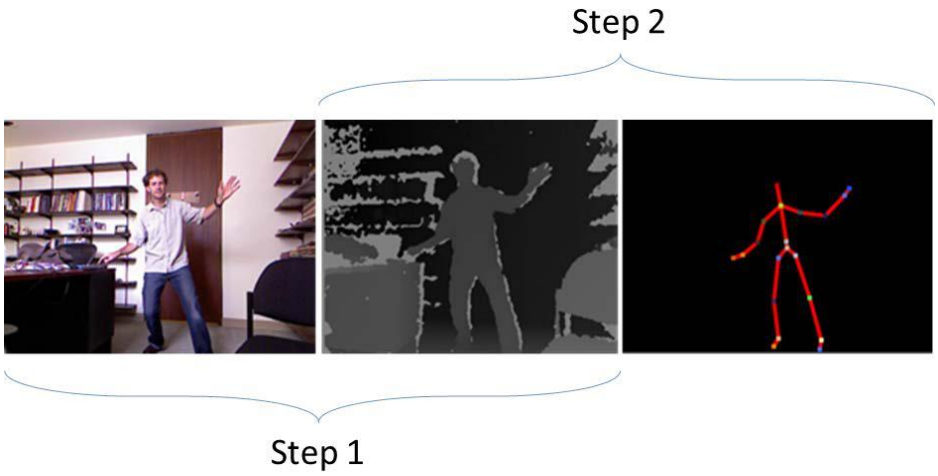


Figure 18: kinect image processing example. Starting from the left: RGB image, the corresponding Depth-Map (center) and eventually the representation of the stick figure generated after the tracking phase (right).

In the first phase, Kinect obtains 3D information coming from the analyzed scene, thus creating a depth that allows to discriminate objects closest to the Kinect compared to those further away. Each pixel is associated with a specific distance from the camera, representing reality and creating the so-called “depth map”. Specific algorithms allow to quantify this distance and correlate the information with the original video image. Once a depth image has been obtained, the next processing step consists of a tracking operation. The latter is entirely carried out by the software

where the main body segments, together with their position and orientation, are identified.

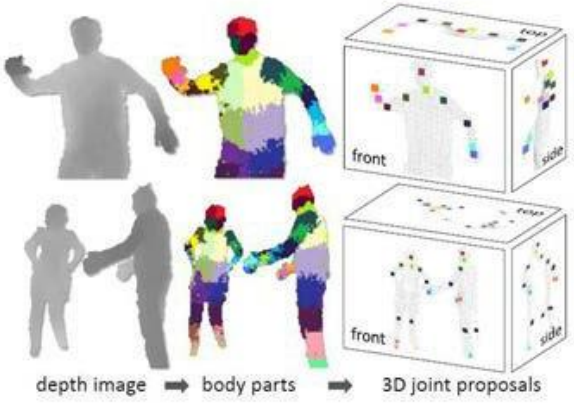


Figure 19: example of Depth Map representation.

Eventually, sensor-based systems do not comprehend cameras, but sensors that rely on the inertia principle. Motion sensors are composed typically by accelerometers (which measure linear acceleration in m/s^2 and they are built on 3 axis), gyroscopes (which measure angular velocities in degrees/sec) and magnetometers (that measure magnetic field strength in uT, namely micro Tesla, or Gauss, since 1 Gauss is equal to 100 uT). Initial sensors are typically called 6-axis IMU (inertial measurements units comprising gyroscopes and accelerometers) or 9-axis IMU (including accelerometer, gyroscope and magnetometer).

Figure 20: inertial Measurement Units (IMUs) can be considered a combination of these three devices: accelerometers, gyroscopes and magnetometers (left). An example of 9-axis IMU (Motion Tracking device) used for smartphones, tablets, wearable sensors, and other consumer markets (right).

Table 5: includes the most used MoCap systems together with their characteristics. Here we compare IMUs (e.g., XsensMVN), marker-based or marker-less (e.g., Kinect) camera systems, their advantages and disadvantages in order to evaluate how each sensors type is appropriate to the different applications.

	Sensors		
	IMUs	Camera-Based	
		Marker-Based	Marker-Less
Accuracy	High (0.75° to 1.5°) [3]	Very high (0.1 mm and 0.5°) [1]; subject to number/location of cameras	Low (static, 0.0348 m) subject to distance from camera
Set up	Straightforward; subject to number of IMUs	Requires time-consuming and frequent calibrations	Usually requires checkerboard calibrations
Capture volumes	Only subject to distance from station (if required)	Varies; up to 15 × 15 × 6 m [1]	Field of view: 70.6° × 60°; 8 m depth range [5]
Cost of installation	From USD 50 per unit to over USD 12,000 for a full-body suit [4]	Varies; from USD 5000 [2] to USD 150,000 [1]	USD 200 [5] per unit
Ease of use and data processing	Usually raw sensor data to ASCII files	Usually highly automated, outputs full 3D kinematics	Requires custom-made processing algorithms
Invasiveness (individual)	Minimal	High (markers' attachment)	Minimal
Invasiveness (workplace)	Minimal	High (typically, 6 to 12 camera systems)	Medium (typically, 1 to 4 camera systems)
Line-of-sight necessity	No	Yes	Yes
Portability	Yes	Limited	Yes
Range	Usually up to 20 m from station [3] (if wireless)	Up to 30 m camera-to-marker [1]	Low: skeleton tracking range of 0.5 m to 4.5 m [5]
Sampling rate	Usually from 60 to 120 Hz [3] (if wireless)	Usually up to 250 Hz [1] (subject to resolution)	Varies; 15–30 Hz [5] or higher for high-speed cameras
Software	Usually requires bespoke or off-the-shelf software	Requires off-the-shelf software	Requires bespoke software, off-the-shelf solutions not available
Noise sources and environmental interference	Ferromagnetic disturbances, temperature changes	Bright light and vibrations	IR-interference with overlapping coverage, angle of observed surface
Other limitations	Drift, battery life, no direct position tracking	Camera obstructions	Camera obstructions, difficulties tracking bright or dark objects
Favoured applications	Activity recognition , identification of hazardous events/poses	Human–robot collaboration , robot trajectory planning	Activity tracking , gesture or pose classification

[1] Based on a sample layout with 24 Prime^{x}41 Optritrack cameras. [2] Based on a sample layout with 4 Flex 3 Optritrack cameras. [3] Based on the specs of the Xsens MTW Awinda. [4] Based on the Xsens MVN. [5] Based on the Kinect V2.

The first MOCAP system installed on board of ISS was ELITE-S2 (Ferrigno *et al.*, 2003), but recently NASA managed to install a new heir: the BTS-Smart system, previously mentioned. The same device is also available in Politecnico di Milano and will allow to obtain useful information through the comparison of data acquired on ISS with the ones collected on ground.

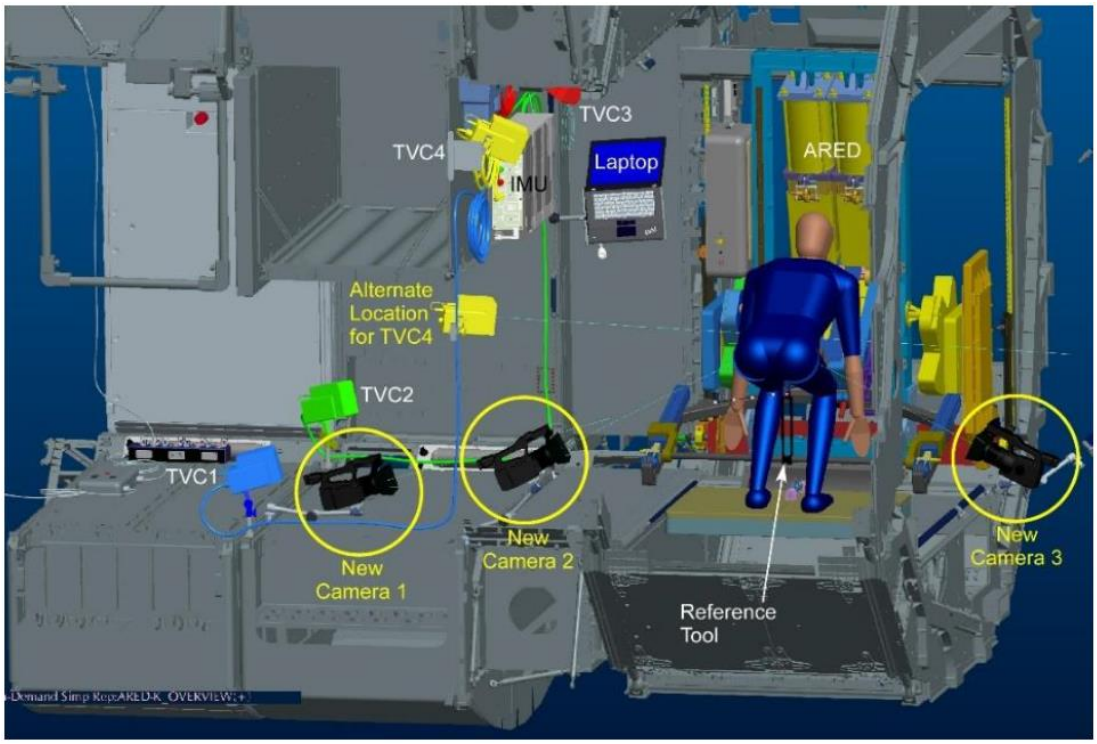


Figure 21: previous Elite-S2 camera placement (TVC1, TVC2, TVC3, TVC4) and available HD cameras (New Camera 1, New Camera 2, New Camera 3). (Ravizza, 2018)

Also, NASA programmed to install a new toilet in node 3. In February 2019, the Toilet Stall hardware was deployed on ISS (only the first half of the project could be completed at this time). The modifications implied many changes both for the installation of the new BTS-Smart system and ARED. The second half of the toilet later reached the ISS in October 2020. Eventually, the final goals regard the introduction of IMU sensors to be used with ARED on ISS. These improvements will help NASA preparing for future missions, including those to the Moon and Mars.

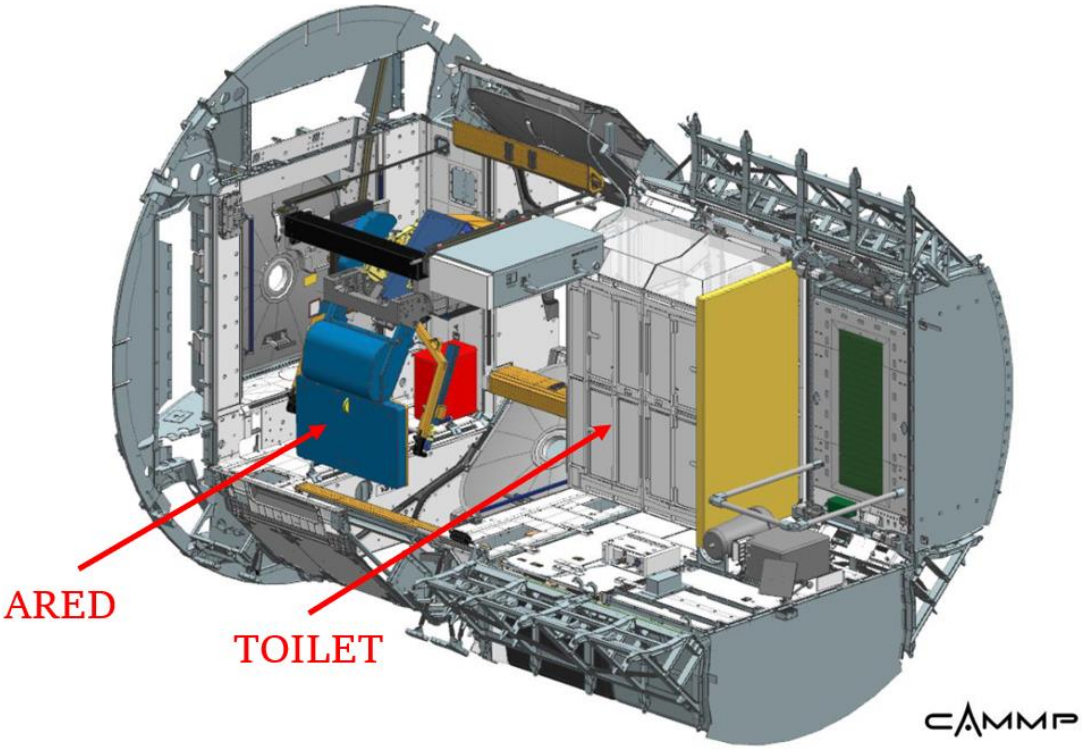


Figure 22: layout of Node 3 with ARED and the Toilet Stall Deployed (Borrego et al., 2019)

Figure 23: toilet Stall deployed in ISS Node 3 (Borrego et al., 2019). The second half of the toilet reached the ISS in October 2020.Its features and improvement will help NASA preparing for future missions, including those to the Moon and Mars.

CHAPTER 2
AIM OF THE THESIS

This thesis is inserted in the research project "MARcatori biologici e funzionali per la biomedicina aStronautica di PREcisione – MARS-PRE", proposed by Italian Space Agency (ASI) and structured in 21 Work Packages (WPs). The characterization of human adaptation to gravity conditions different from Earth can be conducted through the analysis of biological markers (which are the main goals of MARS-PRE project). Biochemical measures of fluids, structural and physical properties of tissues, functional test of muscular strength and biomarkers of sensory-motor behavior are typical example. Nodo 1700 has been assigned to Politenico di Milano, whose WP is entitled: "Marcatori personalizzati del programma di contromisure attraverso machine learning e sensori indossabili per biofeedback".

The purpose is to design a system able to monitor astronauts' workout executed with ARED on ISS. Cosmonauts, indeed, perform a set of exercises without any online or offline trainer's help. To solve this issue, anticipatory indicators of possible training inefficacy and/or injury could be evaluated starting from parameters extracted from a small set of inertial sensors (simulated and real). Consequently, it is important to choose a specific set of devices reusable even in microgravity conditions. To reach this aim, instead of depending on the usual kinematic variables of joint and angular positions, the project relied on acceleration and gyroscopic measures.

Both a biomechanical model including muscular actions and a machine learning algorithm are required to reach the aforesaid scope. The model itself must be inserted

in a virtual environment where target exercises, proposed on ISS, can be simulated both in Earth gravity and microgravity conditions. A machine learning system will then associate the risk level with measurements coming from accelerometers and gyroscopes. In this way, the astronauts will be warned in real time about any incorrect exercise execution. The biofeedback provided will be useful to avoid atrophy, musculoskeletal damage and deconditioning, bone demineralization, cardiovascular deconditioning and coordination defects (important factors when considering long-term missions including Moon or Mars expeditions).
The whole setup could interest even other applications on Earth: rehabilitation (to allow continuity of care with reliable and simple technologies) and athletes training (to optimize their performance).

The biomechanical model would be also used in the contest of project "ARED Kinematics – Biomechanical quantification of bone and muscle loading to improve the quality of microgravity countermeasure prescriptions for resistive exercise", which involves ESA, NearLab at Politecnico di Milano, NASA JSC and Kayser Italia. The aims can be listed as follows: investigate internal forces to optimize resistance exercises prescriptions when using ARED, maximizing their effectiveness and minimizing the required time. Kinematics and dynamics data will, hence, be collected from each astronaut during pre-, in- and post-flight conditions with three different load levels.

Our work was focused on the following aims:

- The primary objective refers to the improvement of a model useful to conduct a biomechanical analysis of the human motion in microgravity conditions on ISS. The OpenSim model employed in the current project allows to reach higher ranges of motion with respect to the one previously considered (Ravizza et al., 2020).
- The secondary goal regards the acceleration signals extraction from correct and incorrect exercises in specific body points. The measurements afore mentioned want to reproduce the same output obtainable from inertial sensors placed on

these anatomical landmarks. The collected data will become the input of machine learning algorithms (e.g., Convolutional Neural Networks, Support Vector Machines, Decision Trees, K-Nearest Neighbor, Extreme Gradient Boosting, Multi-Layer Perceptron), in order to discriminate in real time correct and incorrect exercise executions. Consequently, these body points will be identified as anticipatory biomarkers of muscle-skeletal damage.

- Once chosen the most suitable ML algorithm (starting from the comparison of the 6 ones used in this work), the latter will be exploited to obtain a customized classifier for each single astronaut.

Up to now, we verified the validation of OpenSim model by using data collected from 6 subjects, who performed target exercises using ARED at the NASA JSC. The ML classifier was trained, instead, with acceleration data coming from 17 volunteers, who executed the same target exercises (correctly and incorrectly) using barbell and weights (load within 50-75% of ISO-MAX), according to a protocol approved by the Etic Committee of Politecnico di Milano and ASCR NASA JSC.

The following page shows the conceptual scheme which visually explains the current project. Considering the fact that no IMU sensors are currently available on the ISS, another approach has been taken into account. The new MOCAP system installed between the end of 2020 and the beginning of 2021 will allow to track the astronauts' motion during their trainings with ARED. This system, namely BTS SMART DX, tracks the position of passive markers located on the subjects' body. Via OpenSim software, a biomechanical model will guarantee to simulate IMU placement on body and extract analogues (acceleration) data. The platform enables also to simulate the exercise in the same gravity conditions found on the ISS. The puzzle is completed with the integration of a machine learning system, able to acquire the inertial information previously mentioned. The input data will be processed by the algorithm, thus providing in real time a biofeedback to the cosmonauts. In this way, the latter will know whether they are performing correctly a specific exercise (e.g. Normal Squat, Wide Squat, Deadlift) or not. In case of incorrect execution, the system will also provide information about which type of mistake was made.

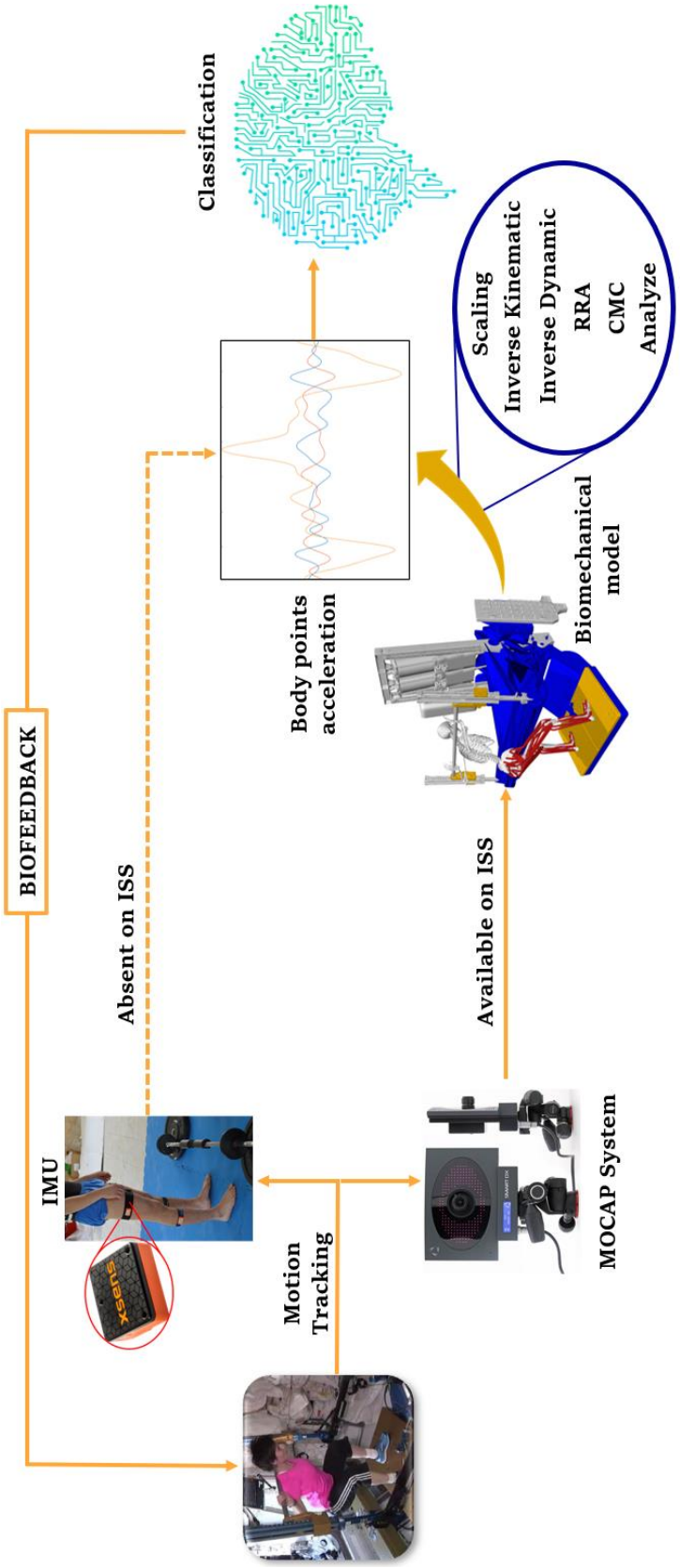


Figure 24: thesis workflow.

CHAPTER 3

OPENSIM 4.1 ®

In recent years, thanks to a rapid development of software engineering, it has been possible to create an open-source simulation environment, namely OpenSim, which allows researchers to share and integrate multiple and different dynamic simulations. In the following chapter, we will describe a brief history of the software origin, the software itself, its potential, the types of data it accepts as input and the various embedded Tools. The latter will be then explained, in order to have a better comprehension regarding the software capabilities.

In the early 1990s, Delp and Loan, at the National Center for Simulation in Rehabilitation Research (NCSRR) at Stanford University, introduced musculoskeletal modeling software, called SIMM (Software for Interactive Musculoskeletal Modeling), which allows users to create, modify, and evaluate models of different structures of the musculoskeletal system. Using SIMM, models of the lower and upper limbs were developed. These progresses allowed to obtain different outcomes: examine the biomechanical consequences of surgical interventions (such as osteotomies and joint prosthesis grafting); estimate the length of the muscle-tendon complex; estimate the arms of the moments of individual muscles with respect to a joint; calculate the speed and accelerations induced and the forces present on the knee during a movement. Moreover, the software also provides the possibility of calculating muscle activations and forces and presents several tools that allow the analysis of the results of dynamic simulations.

As far as OpenSim is concerned, it is an open-source platform managed on Simtk.org by a group of researchers. It is used for modeling, simulation and analysis of the neuro-musculoskeletal system. This software consists of low-level computational tools that are called up by an application. OpenSim is written in ANSI C ++ and the graphical interface (GUI), written in Java, allows users to develop, analyze and visualize models of the musculoskeletal system and generate dynamic movement simulations. The software can be used on all common operating systems.

An OpenSim model represents the dynamics of a system made up of rigid bodies and joints on which forces act to produce movement. The .osim file (which represents the model in OpenSim) consists of components corresponding to parts of the physical system. Most of the properties regarding an OpenSim model can be changed in the GUI. The models of the musculoskeletal system grant the possibility to study neuromuscular coordination, analyze athletic performance and estimate musculoskeletal loads. The muscles stretch over the joints and they generate forces and movements. Once the musculoskeletal model has been created, OpenSim allows users to study the effects of musculoskeletal geometry, joint kinematics, and musculoskeletal properties on joint forces together with the moments that muscles can produce.

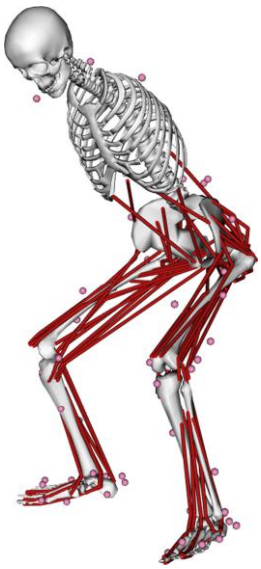


Figure 25: example of OpenSim model displayed via software.

3.1 Features

OpenSim allows the following operations:

- Resizing the size of the musculoskeletal model.
- Inverse kinematics analysis to calculate joint angles starting from marker positions.
- Performing Reverse Dynamics analysis to calculate joint moments, using both joint angles and external forces.
- Solve a direct dynamics problem and generate motion simulations.
- Analyze the data obtained.
- Plot the results obtained.
- Create videos or take snapshots of model movement.

The software features allow you to perform biomechanical simulations and a detailed analysis (using experimental data) of the musculoskeletal system during motor gestures. In this way, it is possible to validate, or not, the model.

The fundamental elements for the simulation are:

- Musculoskeletal model.
- Kinematics data (marker trajectories).
- Experimentally measured reaction force data (Ground Reaction Forces, GRF).

The files that are used to generate a simulation in Opensim are shown below.

3.2 File marker (.trc)

The .trc (Track Row Column) file format was created by Motion Analysis Corporation to specify the position of the markers placed on the subject at different instants of time during the execution of a movement acquired through

stereophotogrammetry. The first three lines of a .trc file consist of a header, followed by two lines containing the names of the markers that must be equal to the names of the virtual markers positioned on the model. Then a blank line is followed by the lines containing all the data relating to the acquisition. Each row of data will contain the frame number of that data, the instant in time and the position coordinates (x-y-z) of each marker. As regards the header, its contents can be listed as follow: 'DataRate' parameter indicates the frequency in Hertz at which the data was acquired; 'NumFrames' corresponds, instead, to the number of data frames; the 'NumMarker' refers to the number of acquired markers and, eventually, 'OrigDataStartFrame' indicates the number of the first frame. Considering a situation in which the .trc file hence generated does not respect the above format, then the data in it contained will not be read correctly by the Opensim software.

PathFileType	4	(X/Y/Z)	C:\POLITECNICO\tesi\Opensim\NS\transformed_S1111NS.trc							
DataRate	CameraRate	NumFrames	NumMarkers	Units	OrigDataRate	OrigDataStartFrame	OrigNumFrames			
250.0	250.0	7341	54	mm	250.0	1	7341			
Frame#	**Time**	LBHD			RBHD			LFHD		
		X1	Y1	Z1	X2	Y2	Z2	X3	Y3	Z3
1	0.0	24,1293	751,7083	22,5822	37,5047	752,0399	96,3049	119,5529	734,5718	19,1921
2	0.004	27,8754	868,6943	26,1015	43,3713	869,0678	111,2919	138,1520	848,8955	22,1763
3	0.008	31,4447	980,2651	29,4594	48,9749	980,6754	125,5855	155,8866	957,9291	25,0217
4	0.012	34,7346	1083,2280	32,5601	54,1555	1083,6680	138,7766	172,2484	1058,5520	27,6469
5	0.016	37,6614	1174,9730	35,3249	58,7818	1175,4360	150,5312	186,8222	1148,2160	29,9853
6	0.02	40,1650	1253,6310	37,6976	62,7590	1254,1090	160,6099	199,3105	1225,0910	31,9895
7	0.024	42,2124	1318,1570	39,6466	66,0332	1318,6420	168,8789	209,5474	1288,1590	33,6325
8	0.028	43,7974	1368,3480	41,1657	68,5924	1368,8320	175,3124	217,5008	1337,2200	34,9094
9	0.032	44,9391	1404,7830	42,2718	70,4636	1405,2600	179,9844	223,2633	1372,8400	35,8349
10	0.036	45,6781	1428,7060	43,0023	71,7072	1429,1710	183,0546	227,0335	1396,2350	36,4408
11	0.04	46,0710	1441,8660	43,4094	72,4085	1442,3130	184,7464	229,0900	1409,1110	36,7719
12	0.044	46,1849	1446,3230	43,5546	72,6695	1446,7510	185,3240	229,7616	1413,4840	36,8808
13	0.048	46,0904	1444,2590	43,5035	72,5980	1444,6660	185,0665	229,3960	1411,4840	36,8232
14	0.052	45,8561	1437,7970	43,3195	72,2997	1438,1840	184,2458	228,3316	1405,1860	36,6532
15	0.056	45,5441	1428,8510	43,0600	71,8702	1429,2200	183,1067	226,8735	1396,4610	36,4199
16	0.06	45,2061	1419,0190	42,7730	71,3895	1419,3720	181,8538	225,2764	1386,8680	36,1640
17	0.064	44,8815	1409,5200	42,4950	70,9193	1409,8590	180,6429	223,7346	1377,6000	35,9166
18	0.068	44,5969	1401,1780	42,2507	70,5018	1401,5080	179,5796	222,3795	1369,4630	35,6989
19	0.072	44,3670	1394,4500	42,0539	70,1607	1394,7730	178,7221	221,2837	1362,9010	35,5224
20	0.076	44,1962	1389,4740	41,9088	69,9041	1389,7950	178,0881	220,4693	1358,0500	35,3908

Figure 26: example of .trc file.

For the realization of this report, the files containing the trajectories of the markers during the static acquisition of the subject and those describing the kinematics of movement were created in this format: the former will be useful in the Scaling phase of the model while the others will guide the model when calculating the inverse kinematics.

3.3 File .mot

The .mot format consists of two main parts: a header and the section corresponding to the data. The former consists of a first line starting that starts with the .mot file name followed by its format. Subsequent rows contain ‘nRows’ (namely the total number of data rows) and ‘nColumns’ (hence the total number of columns). Then, we have the time interval range in which the file data extends. Consecutively, the data is shown in columnar format: the first column represents the instant of time to which each data correspond.

```
S1111NS.mot
version=1
nRows=7339
nColumns=19
inDegrees=yes
endheader
time  R_ground_force_vx R_ground_force_vy R_ground_force_vz R_ground_force_px
      0.00000000      -14.79358307      303.13012700      33.37711692
      0.00400000      -14.72669734      303.73159790      33.54002874
      0.00800000      -14.33689986      305.77740480      33.36529008
      0.01200000      -14.29305900      306.37442020      33.42738040
      0.01600000      -14.10374524      305.18807980      33.45415576
      0.02000000      -13.87398292      305.34024050      33.37994218
      0.02400000      -13.67577072      305.11917110      33.38932696
      0.02800000      -13.50123309      305.13330080      33.35334167
      0.03200000      -13.31709251      305.05590820      33.30946192
      0.03600000      -13.15758543      304.98068240      33.26131114
      0.04000000      -13.02728227      304.88272090      33.21105225
      0.04400000      -12.92550214      304.79684450      33.16396819
      0.04800000      -12.86432061      304.74465940      33.13641413
```

Figure 27: example of .mot file.

The labels contained .mot file (as shown in figure 27) are:

- Time;
- Reaction forces along X, Y, Z (vx, vy, vz);
- Point of application (px, py, pz);
- Pairs (x, y, z).

Every row contains the corresponding data (referred to the reference system of the model) for each instant of time in which the movement was sampled. The files created with this format are those containing the reaction forces (in the three x-y-z components) expressed with respect to time.

3.4 Opensim workflow

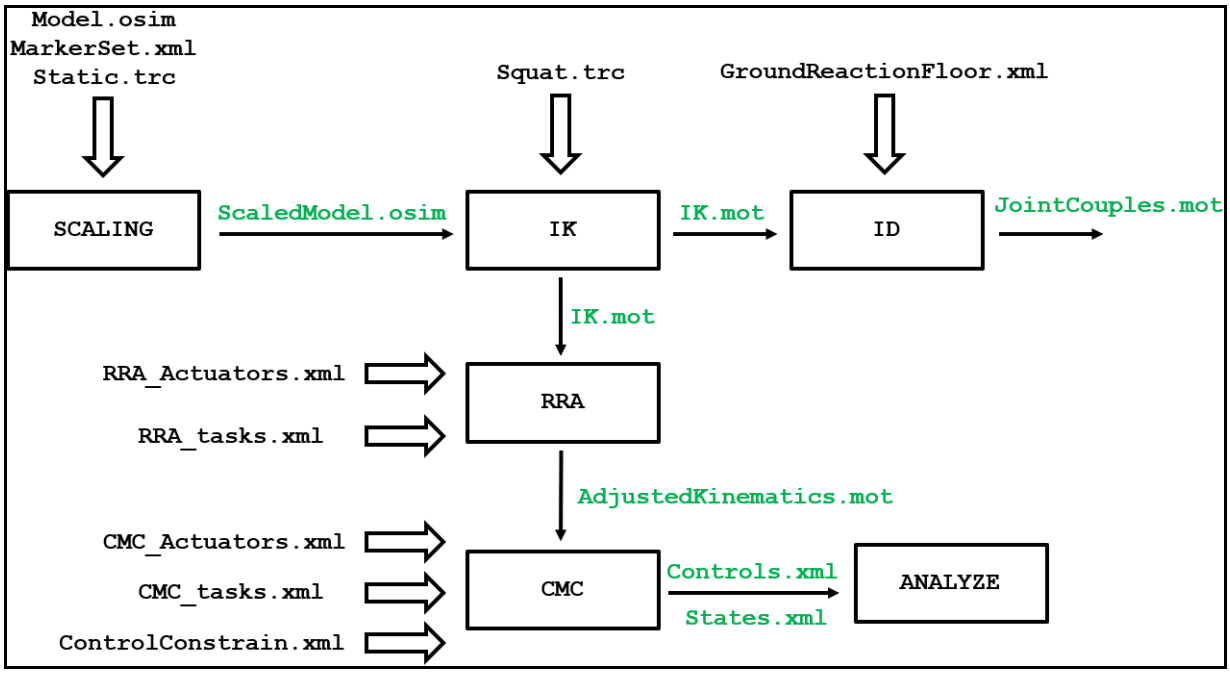


Figure 28: brief OpenSim workflow.

OpenSim includes robust tools to execute biomechanical simulations of musculoskeletal system during motor gestures and to analyze them. Among all the tools, those relevant for this specific project regard model scaling, inverse kinematics (IK) and inverse dynamics (ID) problems resolution, using the Analyze tool to extract virtual markers trajectories (useful to obtain their acceleration), plotting results, creating video and capturing screenshots of model during simulation.

The following steps are needed to analyze experimental data collected in laboratory:

- Preparing kinematics and dynamics data in specific formats.
- Choosing/creating a model and putting virtual markers on it, in order to reproduce.
- The same marker set-up used during data acquisition.
- Scaling the model basing on anthropometry of every subject.
- Computing the IK problem to obtain joint angles that better reproduce movement captured with the MOCAP system.
- Solving the ID problem to determine joint torques.
- Reducing residuals to minimize errors deriving from modeling, so due to the simplification of the system, and marker data processing. This leads to make generalized coordinates more consistent with GRFs and joint moments computed.
- Extraction of trajectories using the Analyze tool, in order to obtain in post processing velocities and accelerations.

3.5 Scaling

The first operation to be performed to make a correct simulation is that of Scaling the musculoskeletal model which will be done basing on the anthropometry of the subject under examination. Firstly, a static experimental acquisition of the marker positions previously placed on the individual in specific anatomical reference points (experimental markers) is required. In the same way, virtual markers will be placed on the same points in the model in order to have a direct correlation between the individual and the model. The virtual markers must have the same name as the experimental markers. The dimensions of each body segment in the model are calculated based on the relative distances between the marker pairs, obtained by a Motion Capture system, which are matched to the virtual marker pairs.

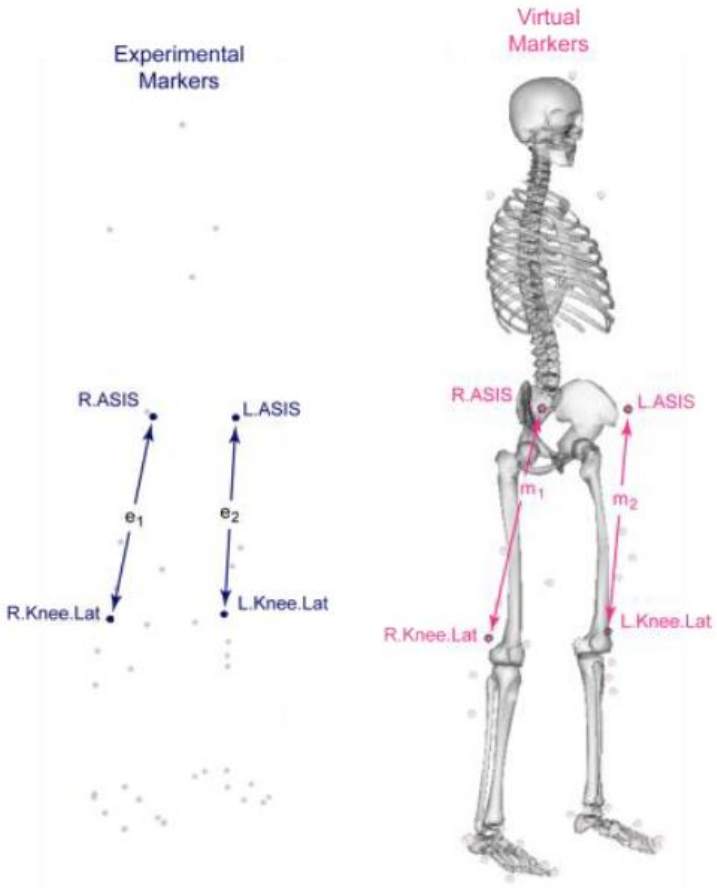


Figure 29: experimental marker positions are computed with MOCAP system (blue markers); virtual markers are placed manually on biomechanical model in anatomical correspondence (pink markers). Distances between experimental markers (e_i) relative to the distances between virtual markers (m_i) are used to compute scale factors.

For example, suppose we use two pairs of markers: p_1 = {R.ASIS, R.Knee.Lat} and p_2 = {L.ASIS, L.Knee.Lat}. The distance for pair 1 on the model (m_1) is calculated by placing the model in its default configuration. The experimental distance for pair 1 (e_1) is obtained by observing each frame of experimental marker data in the given.trc file, calculating the distance between the pair for that frame, and taking the average over all frames over a user-specified time. The scale factor due to torque 1 is therefore $s_1 = e_1 / m_1$. The overall scale factor is the average of the scale factors calculated for all pairs of the same segment (for example, $s = (s_1 + s_2) / 2$ in this case, where s_2 is the scale factor due to pair 2). This global scale factor can then be used

for resizing any segment. However, it is possible (in an alternative way) to scale the body segments with scale factors obtained through anthropometric analysis.

Then, based on the scaling factors, the Scale Tool scales the model geometry, body segments, center of mass location, force application points, and muscle attachment points. For example, the distal attachment point of the soleus muscle is scaled based on the scale factors for the tibial segment. With this operation we also want to modify the masses of the body segments, taking into account the principle that the total sum of the segment's masses (even if redistributed) must be equal to the entire mass of the subject (Preserve Mass Distribution, PMD). Components of a model that depend on distances or lengths, such as ligaments and muscle actuators, are also updated after scaling. For example, muscle's new Optimum_fiber_length and Tendon_slack_length are calculated during the scaling process. The process is complicated by the fact that parameters such as muscle length also depend on the configuration, hence OpenSim tries to keep the model configuration when scaling. Once the Scaling operation has been performed, the virtual markers assume new positions on the musculoskeletal model (more similar to the positions of the experimental markers), so that the subsequent data analysis is as truthful as possible and similar to reality.

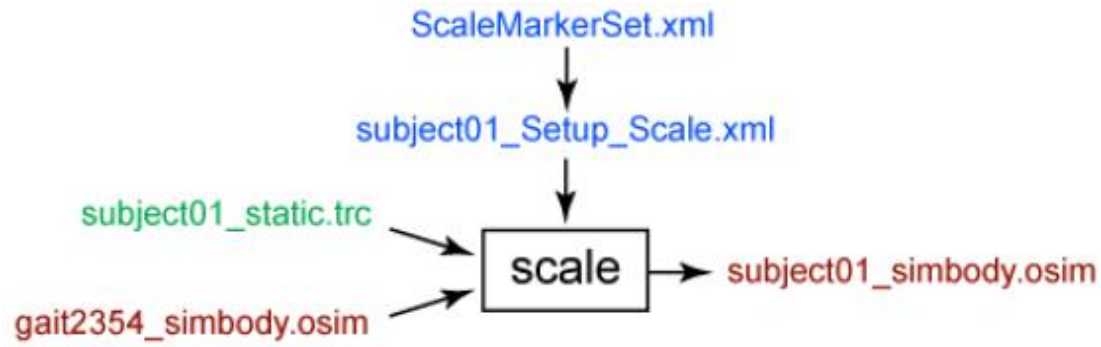
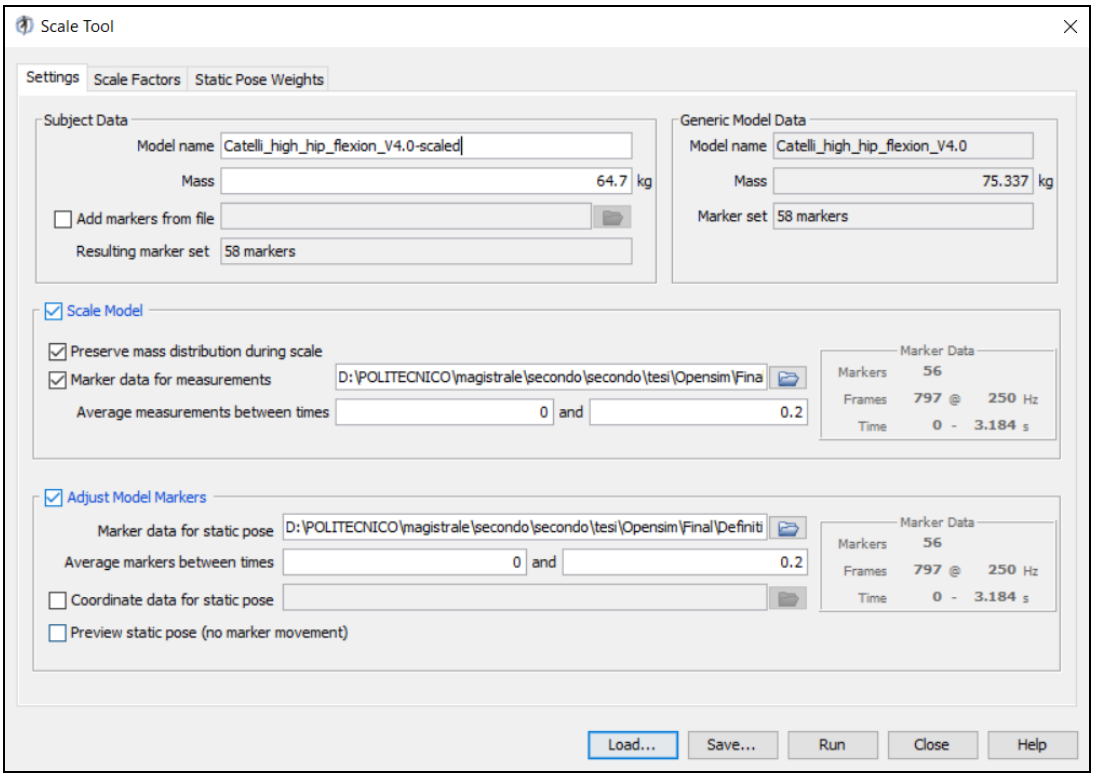
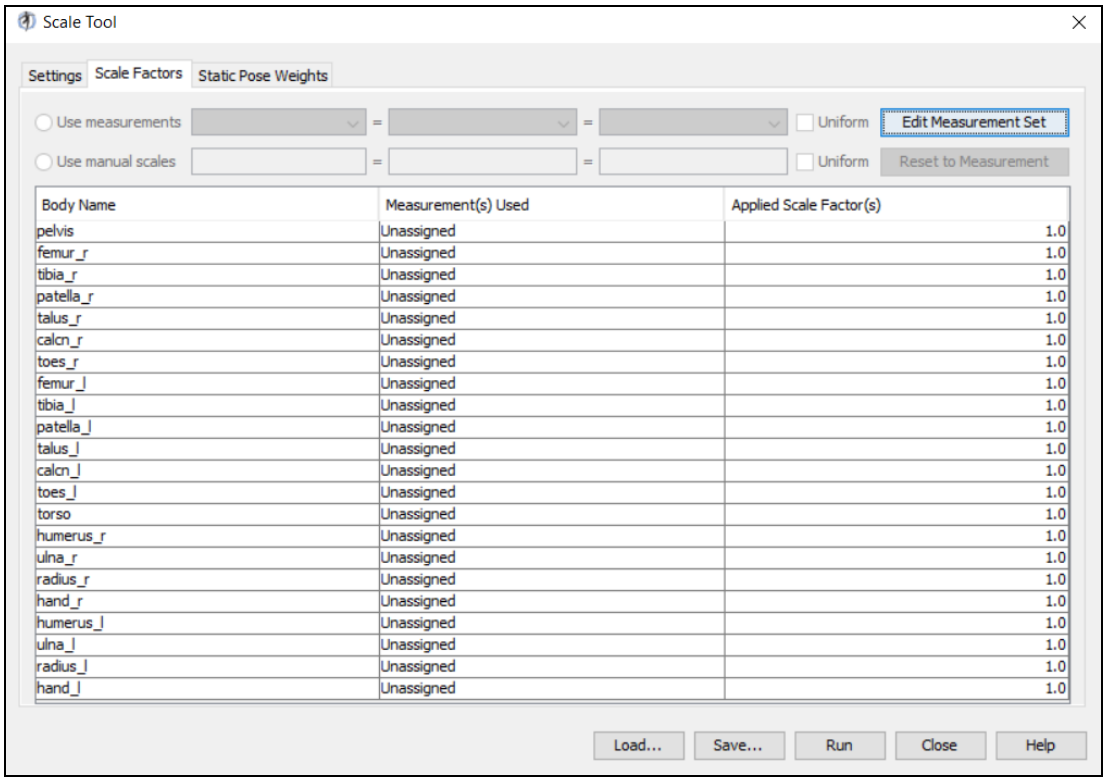


Figure 30: inputs and outputs of the Scale tool. Experimental data are in green, OpenSim files (.osim).





Body Name	Measurement(s) Used	Applied Scale Factor(s)
pelvis	Unassigned	1.0
femur_r	Unassigned	1.0
tibia_r	Unassigned	1.0
patella_r	Unassigned	1.0
talus_r	Unassigned	1.0
calcn_r	Unassigned	1.0
toes_r	Unassigned	1.0
femur_l	Unassigned	1.0
tibia_l	Unassigned	1.0
patella_l	Unassigned	1.0
talus_l	Unassigned	1.0
calcn_l	Unassigned	1.0
toes_l	Unassigned	1.0
torso	Unassigned	1.0
humerus_r	Unassigned	1.0
ulna_r	Unassigned	1.0
radius_r	Unassigned	1.0
hand_r	Unassigned	1.0
humerus_l	Unassigned	1.0
ulna_l	Unassigned	1.0
radius_l	Unassigned	1.0
hand_l	Unassigned	1.0

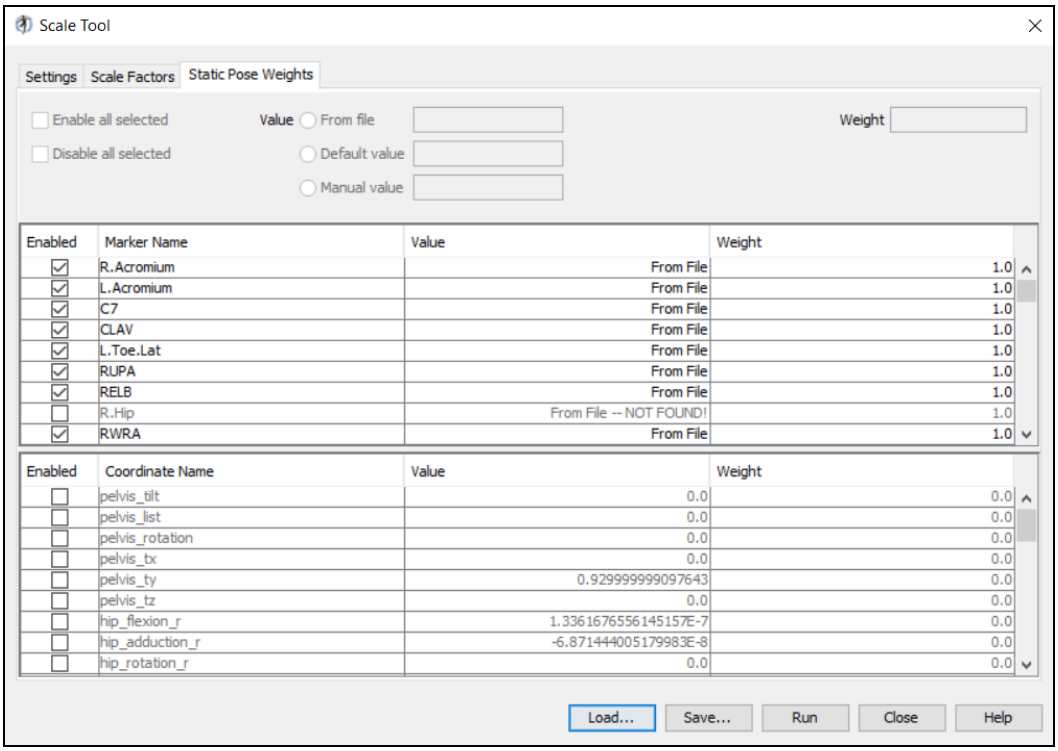


Figure 31: examples of the Scale interface used in the OpenSim Software.

The inputs of the Tool Scale are listed as follows:

- The. osim model to be scaled.
- The .trc file containing the positions of the experimental markers for a static test, i.e. with the subject stationary in a known position.
- The marker set .xml comprising the set of virtual markers positioned on the model segments.

The output hence obtained is the resized model.

3.6 Inverse kinematic

Once the model has been resized, we can proceed with the Inverse Kinematics (IK) procedure. The purpose of this Tool is about finding a set of generalized coordinates (namely the joint angles and the positions of the body segments for the musculoskeletal model) which best reproduce the kinematics of movement referred

to a subject under examination. The Tool IK, for each instant of time, sets the joint coordinates of the model in a position which "corresponds best" to that of the experimental marker and to the coordinate data acquired during that period of time, minimizing the sum of weighted square errors of markers and/or coordinates. The marker error is the distance between an experimental marker and the corresponding virtual marker on the model when its generalized coordinates are those calculated by the IK tool. Each marker has an associated weight that specifies how strongly the error term should be minimized. The coordinate error is the difference between an experimental coordinate value and the coordinate value calculated by the IK instrument. The values of the experimental coordinates can be the joint angles, obtained directly from the Motion Capture system or from an external specialized algorithm (or other measuring devices, such as a goniometer).

A distinction should be made between the 'prescribed' and 'unprescribed' coordinates. The former (also called 'locked coordinate') is a generalized coordinate whose trajectory is known, and which will not be calculated using IK. It will be set to its exact trajectory value. This can be useful when you are confident enough in a generalized coordinate value that you do not want the IK solver to change it. An unprescribed coordinate, instead, refers to a specific coordinate that is not fixed and whose value is calculated using IK. Using these definitions, only the coordinates that are not previously locked can vary and therefore only they appear in the least squares equation solved by IK. Each unprescribed coordinate that is compared to an experimental coordinate must have an associated weight, specifying how strongly the coordinate error should be minimized. Mathematically, the IK tool solves the weighted least squares problem as follows:

$$\min_{q}\left[\sum_{i\epsilon\{markers\}} w_i \left\| x_i^{exp} - x_i(q) \right\|^2 + \sum_{j\epsilon\{unprescribed\ coordinates\}} \omega_i \left\| q_i^{exp} - q_j \right\|^2\right]$$

The instrument finds, for each instant in time, the generalized coordinate vector q which minimizes the cost equation, where x^{exp}_i is the experimental position of the i-th marker, x_i (q) is the position of the corresponding marker on the model, a function of

generalized coordinate values and q^{exp}_j is the experimental value for the j coordinate. All prescribed coordinates are set to their experimental values.

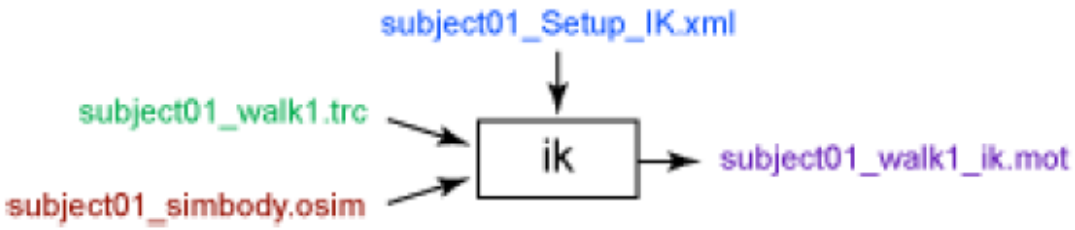


Figure 32: input and output of IK tool. Experimental data in green; OpenSim files are in red; setting file are in blue; output of IK is in purple. Names are as example.

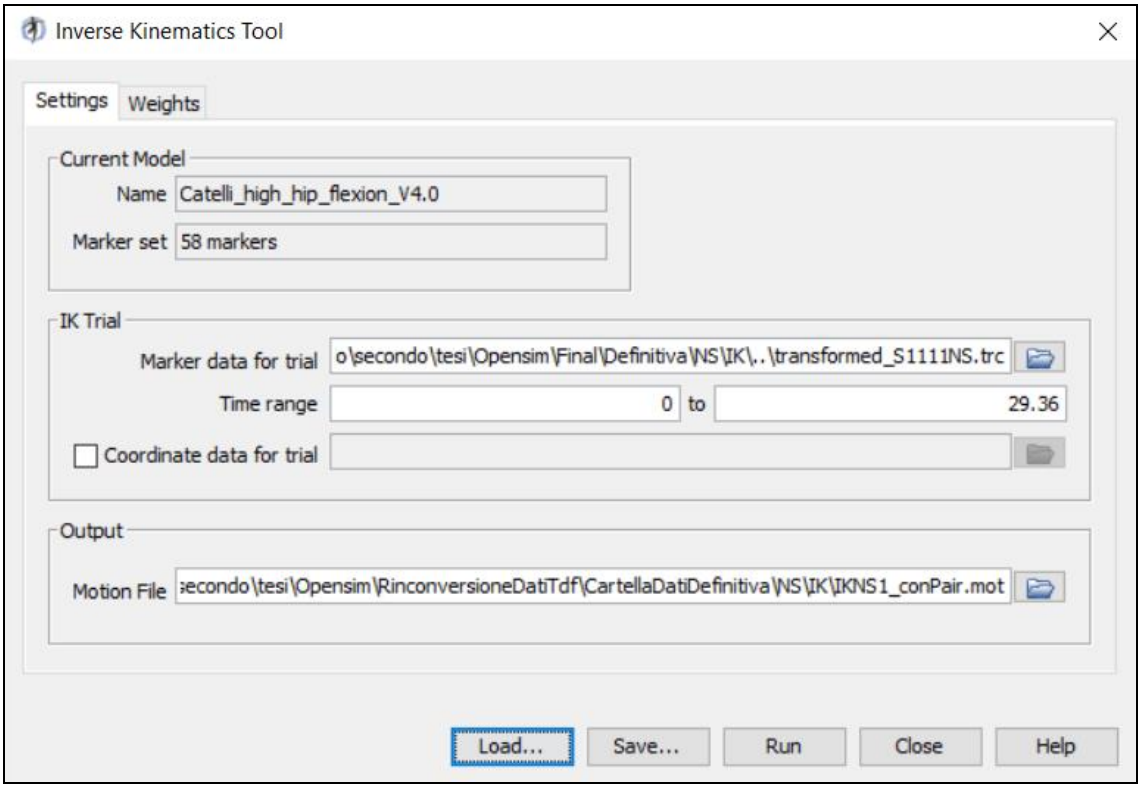


Figure 33: example of the Inverse Kinematic (IK) interface used in the OpenSim Software.

Below, the IK Tool inputs required are listed:

- the file of the marker trajectories obtained thanks to the Motion Capture system related to the entire motor act to be studied.
- the scaled model and the .xml file containing the weight associated with each marker and/or coordinate not described.

As output of the IK, we will have a .mot movement file containing the generalized coordinate trajectories (angles and/or joint translations). The units used by IK are the model units, in particular: meters for length; degrees for angles.

3.7 Inverse dynamic

The .mot file obtained from IK Tool is used as input in the Inverse Dynamics Tool (ID), which determines the generalized forces in each joint responsible for a given movement of the model. Given the kinematics (which describes the movement of the model) and the external loads applied to the model, the Tool ID calculates the internal forces (or torques) generated by the muscles. In classical mechanics, the relationship between force and acceleration is expressed by the Newton's second law (F = ma) by means of equations of motion; ID solves these equations to determine net forces and torques at each joint producing the motion. The classical equations of motion can be expressed in the following form:

$$\boldsymbol{\tau = M(q)\ddot{q} - G(q) - C(q,\dot{q}) - A(q,\dot{q},x,t)}$$

where: for a model with N degrees of freedom, q, $\dot{q}$ and $\ddot{q} \in R^N$ are the vectors of generalized positions, velocities and accelerations; M (q) $\in R^N \times N$ is the mass matrix of the system that depends on the q̇ configuration of the model ; C (q, q̇) $\in R^N$ is the Coriolis vector and centrifugal forces; G (q) $\in R^N$ is the vector of gravitational forces; A (q, $\dot{q}$, x, t) $\in R^N$ is the vector of applied loads, which are the external forces applied to the model (such as ground reaction forces, passive bodies or components active) and which may explicitly depend on time t and on the input controls for the actuators. The resulting generalized forces $\tau \in R^N$ are what the instrument calculates.

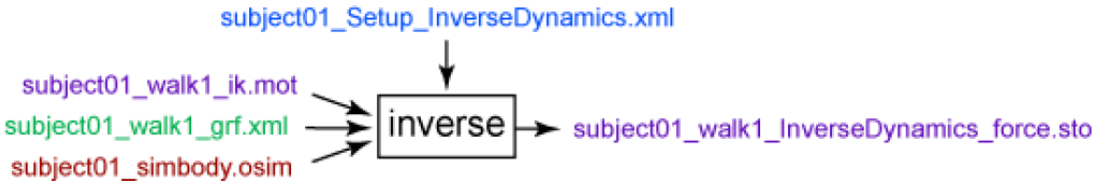


Figure 34: inputs and output of the ID tool. Experimental data in green; OpenSim files are in red; setting file are in blue; output of IK and ID are in purple. Name are as example.

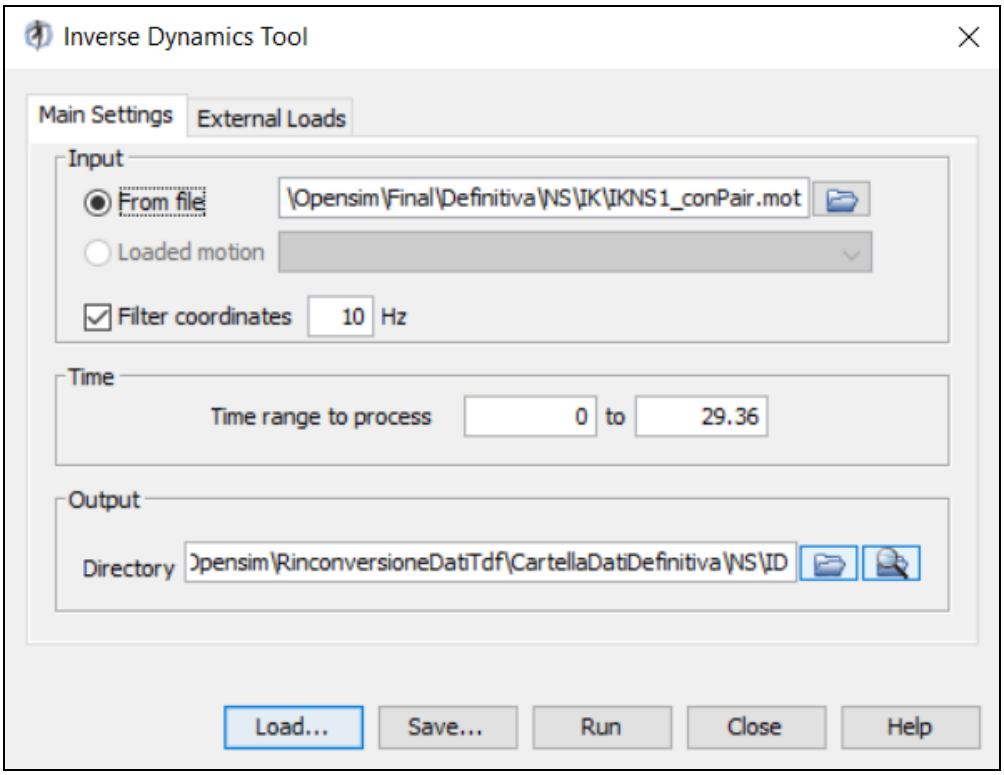


Figure 35: example of the Inverse Dynamics (ID) interface used in the OpenSim Software.

As Input of the ID tool we consider:

- the .mot movement file, containing the temporal histories of the generalized coordinates that describe the movement of the model obtained from IK. It is recommended to check the Filter coordinates item in order to have more homogeneous waveforms.
- As regards the external loads, we set the ground reaction forces (.grf file).

The output is related to an archive file containing the temporal chronologies of the joint couples and forces. In solving the inverse dynamics problem, both kinematic

data and force plate data were used, making this an overly determined problem. In other words, the problem has more equations than unknowns (i.e., degrees of freedom). Due to errors in the experimental movement data and inaccuracies in the musculoskeletal model, it turns out that Newton's second law is violated:

$$F_{exp} \neq m \times a$$

One method of handling this inconsistency is to calculate and apply residual forces and moments to a particular body segment in the model, such that Newton's second law becomes:

$$F_{exp} + F_{residual} = m \times a$$

However, this procedure is intended exclusively for the study of walking where there is a continuous displacement of the center of mass and therefore a continuous imbalance of forces and moments.

The residues are usually applied to the pelvic segment. In the absence of an error between the experimental data and the model data, the residual force must be zero. In practice, however, this is never verified (you can see the residuals from the inverse dynamics solution, plotting pelvis_tx_force, pelvis_ty_force as a function of time).

3.8 Residuals reduction algorithm

The purpose of the Residual Reduction Algorithm (RRA) is to minimize the effects of modeling errors and marker data processing that aggregate and lead to large non-physical compensatory forces called residuals. More specifically, residual reduction alters the center of mass of the torso (or pelvis) of a subject-specific model and allows the kinematics of the model to be varied from the inverse kinematics in order to be more dynamically consistent with ground reaction force data. Residual reduction is a form of direct dynamic simulation that uses a controller to follow the kinematics of the model determined by the inverse kinematics. The Computed Muscle Control (CMC) acts as a controller: it can be used to determine a mass and

kinematic distribution of the joints that are more consistent with the ground reaction forces.

We could explain what RRA actually consists of, using an example. Consider a human skeleton model without upper limbs (e.g., gait23 model) made up of ten rigid segments (bones); 17 of the 23 generalized coordinates of the model represent the angles at the joints. Each of these 17 degrees of freedom is operated by a pair. The remaining six generalized coordinates represent the six degrees of freedom (three translational and three rotational) of the pelvis with respect to the ground. To simulate a walk, the six degrees of freedom of the pelvis are represented as a 'six degrees of freedom joint' operated with as many actuators. Each of these 6 pairs is called a residual actuator. It now has 23 degrees of freedom and 23 actuators, that is, exactly, one actuator per degree of freedom. The three residual actuators that operate the three translational degrees of freedom between the basin and the ground are the residual forces indicated with F_x, F_y and F_z. The three rotational degrees of freedom are operated by residual torques (or moments) indicated with M_x, M_y, and M_z. F_x is the force applied along the X axis (forward), F_y is the force applied along the Y axis (vertical), F_z along the Z axis (transverse), M_x is the torque applied to the X axis, and etc. In other words, the six residual actuators would be equivalent to adding a new term that satisfies Newton's second law: $F + F_{residuals} = ma$. To reduce residual forces and moments, residuals are calculated and averaged over the duration of the motion. On the basis of these averages, the algorithm makes changes in the mass parameters of the model, such as the position of the center of mass of the pelvis. To minimize residues, it is important to:

- Make an initial step with predefined inputs, then check for residuals and coordinate errors.
- Reduce stakeout weight on coordinates with low error.
- Reduce maximum residual excitation or optimum actuator force.

The goal of these residual value restrictions is to reduce the need for residuals to a minimum in order to closely follow the desired kinematics (so that motion is

generated solely by internal joint moments). During computerized muscle control (CMC), the next OpenSim step, moments will derive from forces generated by the muscles. In this way, the biomechanical results on the muscle function (obtained as results of CMC) will be closer to reality rather than letting the residues be arbitrarily large. With these restrictions placed on the residuals, the movement of the model will probably be altered since the residuals themselves may not be able to reach the quantities that would result from the inverse dynamics while following the IK kinematics exactly.

REMINDER: a critical aspect for RRA regards the replacement of muscles with only one ideal actuator per coordinate. In the gait2354 example, these correspond to the residuals for the six degrees of freedom of the pelvis and the reserves for all other internal coordinates of the model (joint angles). Hence, each degree of freedom (DOF) in the model should have an ideal torque or force (reserve) actuator. Thus, we have to consider and include the 6 DOFs of the base segment of the model, which are called "residual actuators". In most cases, these ideal actuators are used to replace muscles in the model. The optimal forces are the maximum output of the ideal actuators (torques, linear forces). The applied torque (force) is equal to the optimum force multiplied by the control value.

The remnant to the pelvis must be applied to the resized COM position. Actuator forces are calculated by choosing force and torque values that minimize an objective function. At the end of the simulation, the average value is calculated for each residual actuator. The average values for Mx (residual left-right torque) and Mz (residual forward torque) are used to adjust the center of mass of the trunk to correct the excessive "tilt" of the model due to inaccuracies in mass distribution and in the bust geometry in the model.

In order to evaluate the results obtained, you can consider these tips:

- The RMS difference in joint angle during movement should be less than 2-5º (or less than 2cm for translations).

- Peak residual forces should typically be less than 10-20 N. Average residuals should typically be less than 5-10 N.
- Comparing the residual moments of RRA with the moments of Inverse Dynamics, you should see a 30-50% reduction in residual peak moments.
- Compare the torque/joint forces with those present in literature.

Table 6: example of threshold values used to evaluate RRA results for full body walk and run simulations.

Thresholds:	GOOD	OKAY	BAD
MAX Residual Force (N)	0-10 N	10-25N	> 25 N
RMS Residual Force (N)	0-5 N	5-10 N	> 10 N
MAX Residual Moment (Nm)	0-50 Nm	50-75 Nm	>75 Nm
RMS Residual Moment (Nm)	0-30 Nm	30-50 Nm	>50 Nm
MAX pErr (trans, cm)	0-2 cm	2-5 cm	>5 cm
RMS pErr (trans, cm)	0-2 cm	2-4 cm	>4 cm
MAX pErr (rot, deg)	0-2 deg	2-5 deg	> 5 deg
RMS pErr (rot, deg)	0-2 deg	2-5 deg	> 5 deg

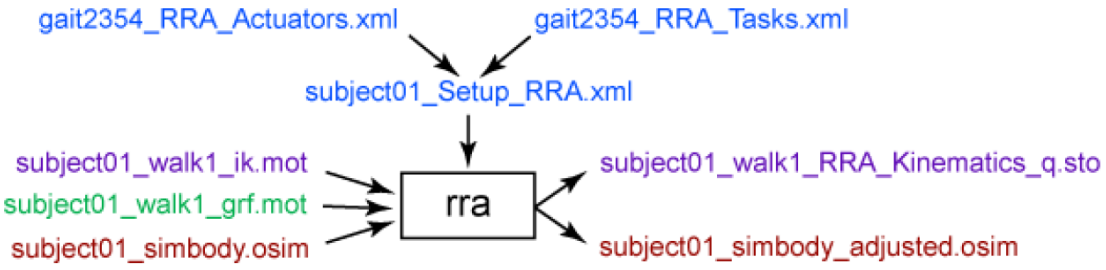


Figure 36: input and output of RRA tool. Experimental data are in green; OpenSim files are in red; setting file are in blue; output of IK and RRA are in purple. Names are as example.

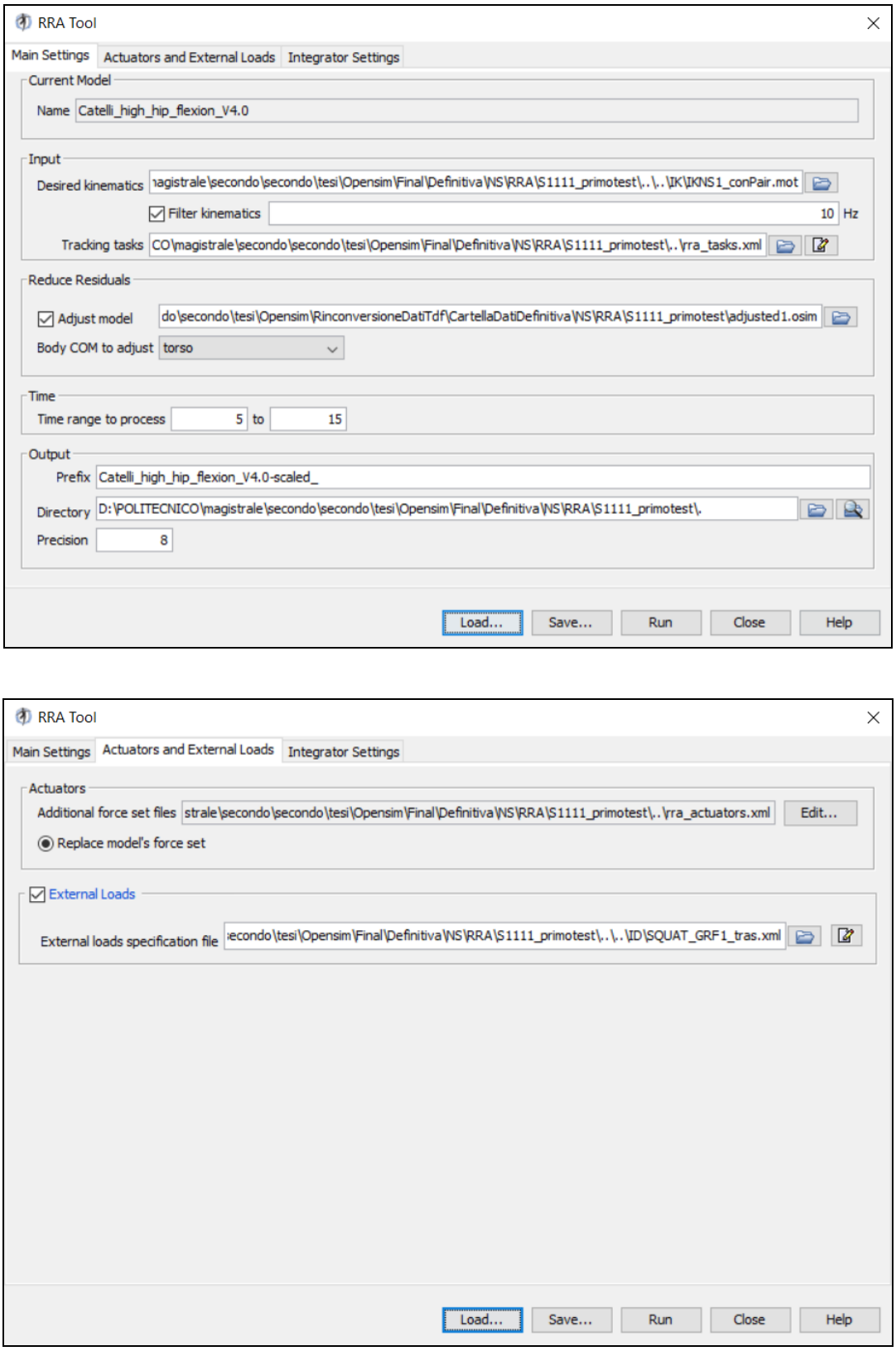


Figure 37: examples of the Scale interface used in the OpenSim Software.

In input to RRA are requested:

- The .mot file obtained from the IK Tool.
- The tasks .xml file which specifies what are the coordinates to plot and the corresponding weight (which determines how well the joint angles follow the angles specified by IK).
- The external loads file grf.mot.
- The file of the joint actuators used to replace the muscles (it contains the name of the generalized coordinates to which the actuator is applied, the minimum and maximum value of the control signal applied and the maximum generalized force produced by it).

After loading the required data, we can start the simulation and, as output, we will obtain a model with adjusted mass properties, actuator excitations (i.e., control signals necessary to generate actuator forces and torques), actuator forces and torques (corresponding adjusted kinematics), joint angles, velocities and accelerations, position errors for each of the generalized coordinates of the model during the test.

3.9 Computerized muscle control

The CMC, Computerized Muscle Control, previously mentioned, is another peculiar Tool embedded in OpenSim. In this case, the aim is to calculate a series of muscle excitations that drive the dynamic musculoskeletal model towards a particular desired kinematics. The CMC method applies this concept by using a proportional-derivative PD control and static optimization. Firstly, the initial states of the model are calculated, i.e. the initial values of the generalized coordinates q (joint angles), the generalized speeds, plus any initial states such as the levels of muscle activation and the length of the fiber at the beginning of the movement. While the initial values of the generalized coordinates and velocities can be taken from kinematics, the initial values of the muscle states are generally unknown. The first step of the CMC

algorithm is to calculate a series of desired accelerations ($\ddot{q}$ *), starting from the experimental generalized coordinates (q_{exp}). The desired accelerations are calculated using the following PD law:

$$\ddot{\boldsymbol{q}}*(\boldsymbol{t}+\boldsymbol{T}) = \ddot{\boldsymbol{q}}_{\boldsymbol{exp}}(\boldsymbol{t}+\boldsymbol{T}) + \boldsymbol{kv}\,[\dot{\boldsymbol{q}}_{\boldsymbol{exp}}(\boldsymbol{t}) - \dot{\boldsymbol{q}}(\boldsymbol{t})] + \boldsymbol{kp}\,[\boldsymbol{q}_{\boldsymbol{exp}}(\boldsymbol{t}) - \boldsymbol{q}(\boldsymbol{t})]$$

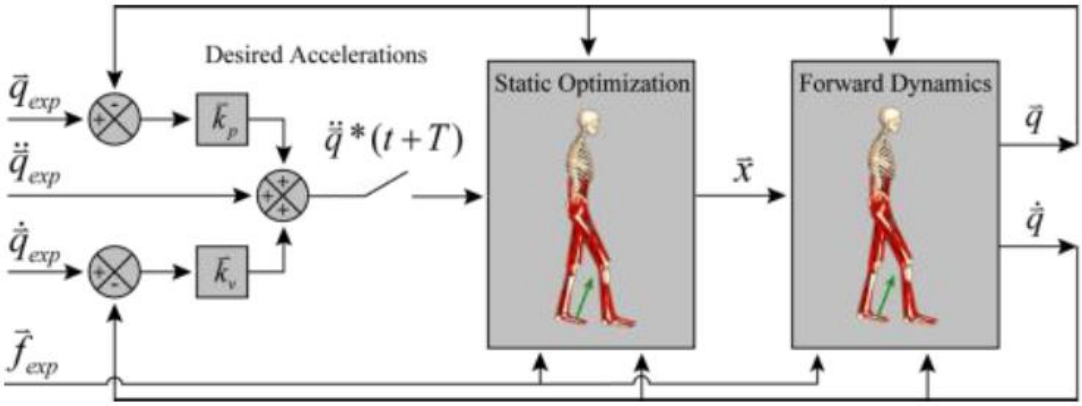


Figure 38: conceptual scheme explaining the CMC algorithm.

Where: $\ddot{q}_{exp}$ and $\dot{q}_{exp}$ are, respectively, the accelerations and velocities of the experimental markers calculated as derivatives in time of their position coordinates; k_v and k_p are the feedback gains on velocity and position errors. Since the forces applied by muscles to the body cannot change instantly, the desired accelerations are calculated for some time T in the future. For musculoskeletal models, T is typically chosen for approximately 0.010 seconds. This time interval is short enough to have adequate control and long enough for the muscle forces to change. If these desired accelerations are obtained, the errors between the model coordinates and the experimentally derived coordinates will be brought to zero. To bring these errors to zero critically, the speed gains can be chosen using the following relationship:

$$\boldsymbol{k_v} = \boldsymbol{2\sqrt{k_p}}$$

For musculoskeletal models, it works well if error gains are chosen in order to slowly bring errors to zero (k_v = 20 and k_p = 100).

The next phase is about considering the static optimization in which actuator controls x(t), which are muscle forces, are obtained to achieve the desired accelerations $\ddot{q}*$ (t + T). The definition of 'static' is referred to the computation of quantities in each time instant. Static optimization works basing on two formulations: the first one (Eq. X), called slow target, minimizes and distributes loads across actuators and the model accelerations toward the desired accelerations; the second equation (Eq. Y), called fast target, is the sum of squared controls augmented by a set of constraints Cj = 0 that require the desired accelerations to be achieved within the tolerance set for the optimizer. The last one is faster and produce better tracking, but if constraints cannot be met, it fails.

$$J = \sum_{i=1}^{n_x} x_i^2 + \sum_{j=1}^{n_q} \omega_j \left(\ddot{q}^*_j - \ddot{q}_j \right)^2 \quad \text{(X)}$$

$$J = \sum_{i=1}^{n_x} x_i^2 \qquad C_j = \ddot{q}^*_j - \ddot{q}_j \quad \forall j \quad \text{(Y)}$$

In the end, CMC computes controls to conduct a standard forward dynamic simulation over time, so that muscle excitations are estimated.

The scheme in figure 39 describes the workflow starting from measured data, passing through IK, ID and CMC to obtain muscle forces.

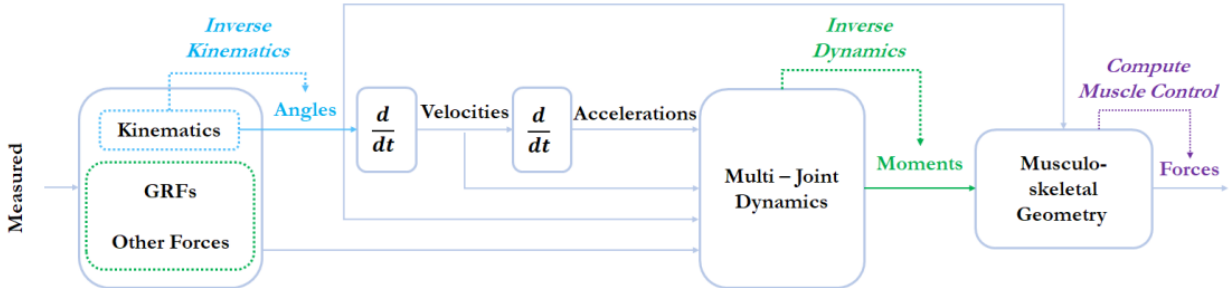


Figure 39: from left to right it is shown the workflow to obtain muscle forces starting from data acquisition using MOCAP system and force plates. Kinematics data measured with MOCAP system (markers' coordinates over time) are in light blue and they are used to compute IK, so that joint angles are obtained. Force data (GRFs and external forces) utilized to resolve the ID problem, which needs also joint angular accelerations obtained by deriving twice joint angles, are in green. In the end, complex algorithms estimate control actuators (muscle forces) using data obtained in the step before.

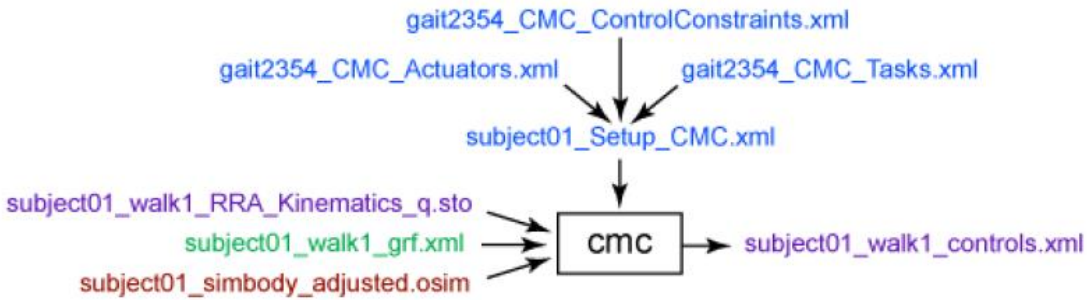
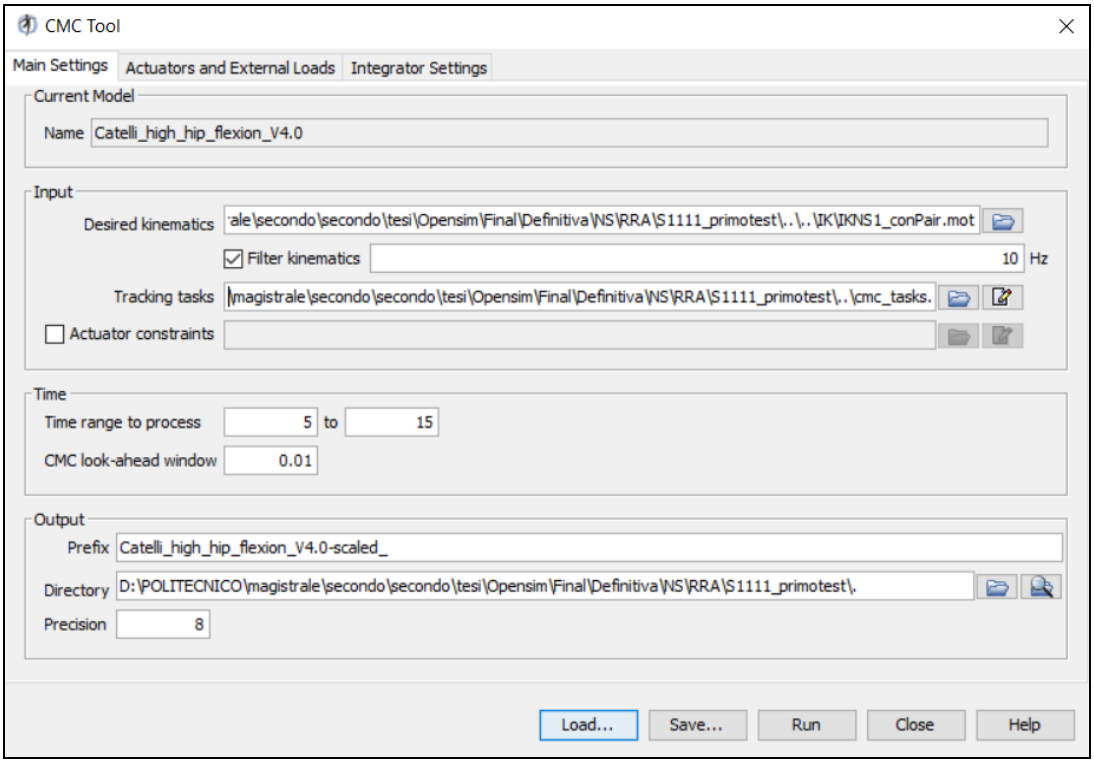


Figure 40: inputs and outputs regarding the CMC tool. Experimental data are in green; OpenSim files are in red; setting file are in blue; output of IK and RRA are in purple. Names are as example.



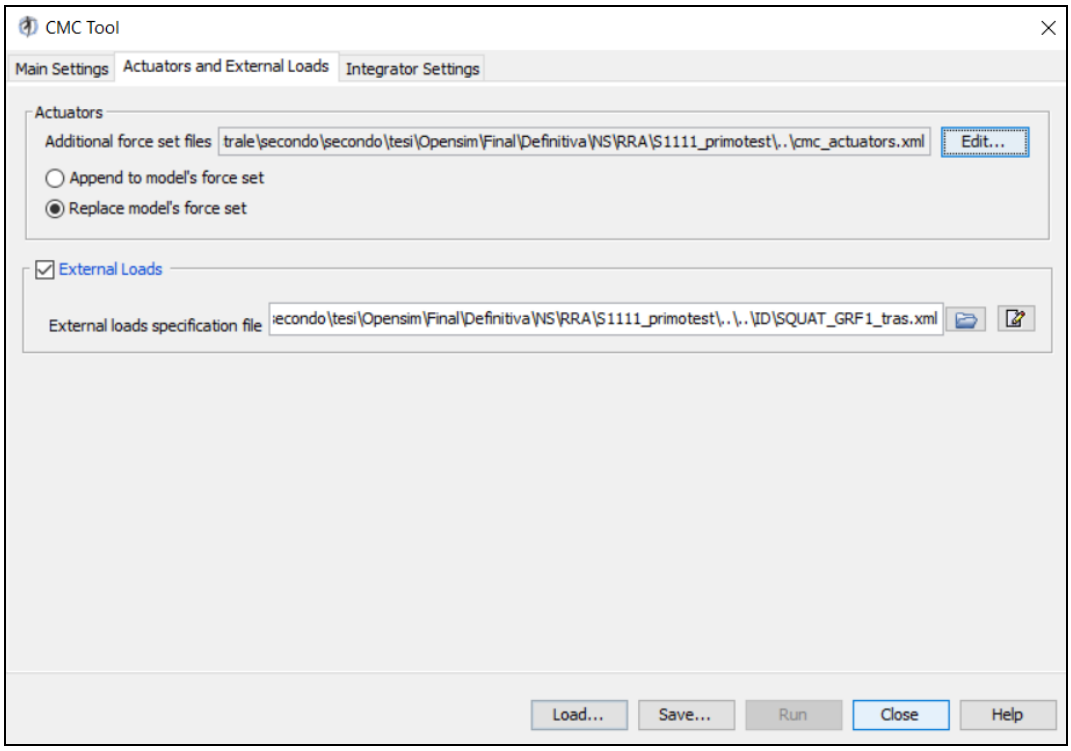


Figure 41: examples of the CMC Tool interface used in the OpenSim Software.

Opening the CMC Tool in OpenSim, as inputs we will have:

- The kinematics coming out of RRA.
- The .xml file that specifies which coordinates to track and the corresponding tracking weight.
- In case it exists, it is important to insert the actuator constraints file which contains the limits on the model actuators, which include muscles, spare and residual actuators. The control constraint file specifies the maximum and minimum "excitation".
- In the ‘Actuators and External Load’ section, the external load data and the residual actuators are needed.

The output of CMC, instead, includes three files:

- Controls.xml file, which contains the excitations for the individual muscles and the controls for any residual and/or reserve actuators.
- Forces.sto file, that contains the reserve and/or residual muscle forces and pairs.

- States.sto file, which contains the model states and muscle states of the simulated movement (i.e., joint angles and velocities, muscle fiber length and activations).

In order to evaluate the results obtained, these tips can be considered:

- Peak reserve actuator torques should typically be less than 10% of the coupling's peak torque.
- Peak residual forces should typically be less than 10-20 N; residual peak moments, instead, less than 75 Nm (depending on the type of movement).
- it is important to compare the simulated activations with the experimental EMG data (recorded by the subject or from the literature). The triggers should show times and entities similar to EMG data.

Table 7: example of threshold values used to evaluate CMC results for full-body walk and run simulations.

Thresholds:	GOOD	OKAY	BAD
MAX Residual Force (N)	0-10 N	10-25N	> 25 N
RMS Residual Force (N)	0-10 N	10-25 N	> 25 N
MAX Residual Moment (Nm)	0-50 Nm	50-75 Nm	>75 Nm
RMS Residual Moment (Nm)	0-30 Nm	30-50 Nm	>50 Nm
MAX pErr (trans, cm)	0-1 cm	1-2 cm	>2 cm
RMS pErr (trans, cm)	0-1 cm	1-2 cm	>2 cm
MAX pErr (rot, deg)	0-2 deg	2-5 deg	> 5 deg
RMS pErr (rot, deg)	0-2 deg	2-5 deg	> 5 deg
MAX Reserve (Nm)	0-25 Nm	25-50 Nm	> 50 Nm
RMS Reserve (Nm)	0-10 Nm	10-25 Nm	> 25 Nm

3.10 Analyze tool

The Analyze tool permits to analyze a model or a simulation basing on several inputs, which can include time histories of model states, controls, and applied external loads. A typical use case is analyzing an existing simulation, which may

have been calculated using the CMC, without having to repeat the simulation. This both saves processing time and allows to analyze a simulation exactly as it happened.

The basic analyzes that can be carried out are listed below:

- Kinematics: it records generalized coordinates (q), generalized velocities (u's) and accelerations (i.e. the derivatives of generalized velocities: du/dt. This is exactly what we used to carry out the current project.
- BodyKinematics: it records the configuration (center of mass position and orientation) of each body, as well as their velocities (linear and angular) and accelerations (linear and angular). It also records the overall center of mass of the model, as well as the speed and acceleration of this center of mass.
- Actuators: it records the generalized force, speed, and power developed by each model actuator. The generalized force can be a force (with unit N) or a torque (with unit Nm). Actuator speed is the speed at which the actuator shortens. Depending on the actuator, a velocity can be a travel speed (m/s) or an angular velocity (degrees/s). An actuator power (Watt) is the speed at which an actuator operates. Positive work means that the actuator is delivering power to the model; negative power means, instead, that the actuator draws energy from the model.

The available analyzes also include, in addition to the three analyzes above mentioned (kinematics, body kinematics and actuators):

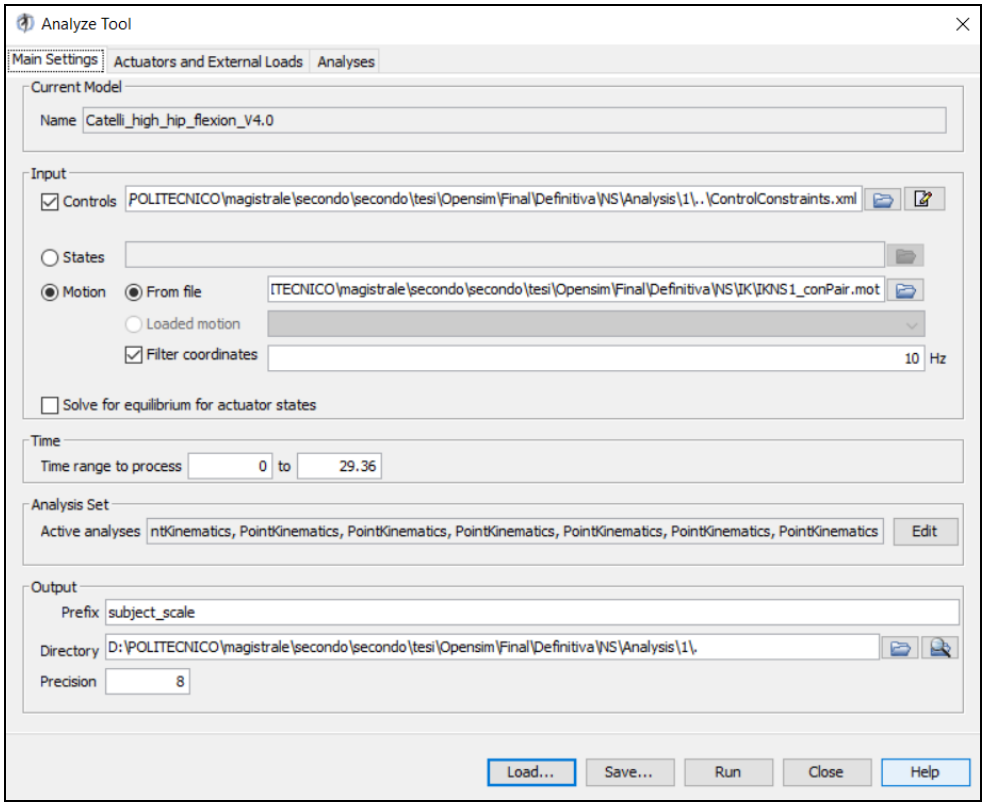
- Joint Reaction Analysis: it reports the joint reaction loads of a model. For a given joint, the reaction load is calculated as the required forces and moments to constrain the movements of the body to satisfy the joint as if the joint did not exist. The reaction load acts on the center of the joint of both the 'parent' and 'child' body and the force can be signaled and expressed in the 'child', 'parent' or 'ground' frame.

Depending on the analysis you want to perform, there are three types of inputs that may be needed to analyze a model:

- States: variables of a model that are governed by differential equations and are therefore integrated during a simulation. The most common examples of states are generalized coordinates (q, e.g., joint angles) and velocities (e.g., joint angular velocities) that specify the configuration of a model. However, states are not limited to coordinates and velocities. Muscles have often been. Muscle activation and fiber length are common examples of muscle states.
- Controls: independent variables used to control the behavior of a model. Muscle excitations are an example. They are not governed by differential equations, but they are generally free to take any value between zero (no excitation) and one (full excitation). Controls in a model are often the variables used as control parameters in optimization problems.
- External Loads: forces or torques applied between the terrain and the bodies of a model. Induced Acceleration Analysis is used to calculate accelerations caused or "induced" by forces acting on a model, such as the contribution of individual muscle forces to the acceleration of the center of mass. Typically, one wishes to study induced accelerations of generalized coordinates (e.g. knee angle) or body positions (e.g. center of mass of the model) and the forces are made up of muscles, gravity and any additional forces (e.g. residual actuators, reserve actuators, etc.).

```
28      <!--Set of analyses to be run during the investigation.-->
29      <AnalysisSet name="Analyses">
30        <objects>
31 ›        <BodyKinematics name="BodyKinematics">
47          <PointKinematics name="PointKinematics">
48            <!--Flag (true or false) specifying whether on. True by default.-->
49            <on>true</on>
50            <!--Start time.-->
51            <start_time>4.5</start_time>
52            <!--End time.-->
53            <end_time>25.5</end_time>
54            <!--Specifies how often to store results during a simulation. More sp
55            <step_interval>1</step_interval>
56            <!--Flag (true or false) indicating whether the results are in degree
57            <in_degrees>true</in_degrees>
58            <body_name>pelvis</body_name>
59            <relative_to_body_name>none</relative_to_body_name>
60            <point_name>V.Sacral</point_name>
61            <point> -0.155 0.035 0</point>
62          </PointKinematics>
63          <PointKinematics name="C7">
64            <!--Flag (true or false) specifying whether on. True by default.-->
65            <on>true</on>
66            <!--Start time.-->
67            <start_time>4.5</start_time>
68            <!--End time.-->
```

Figure 42: example of set-up file for analyzing kinematics of desired points. Circled in black, specification of selected point on pelvis whose coordinates relative to global reference frame are specified.



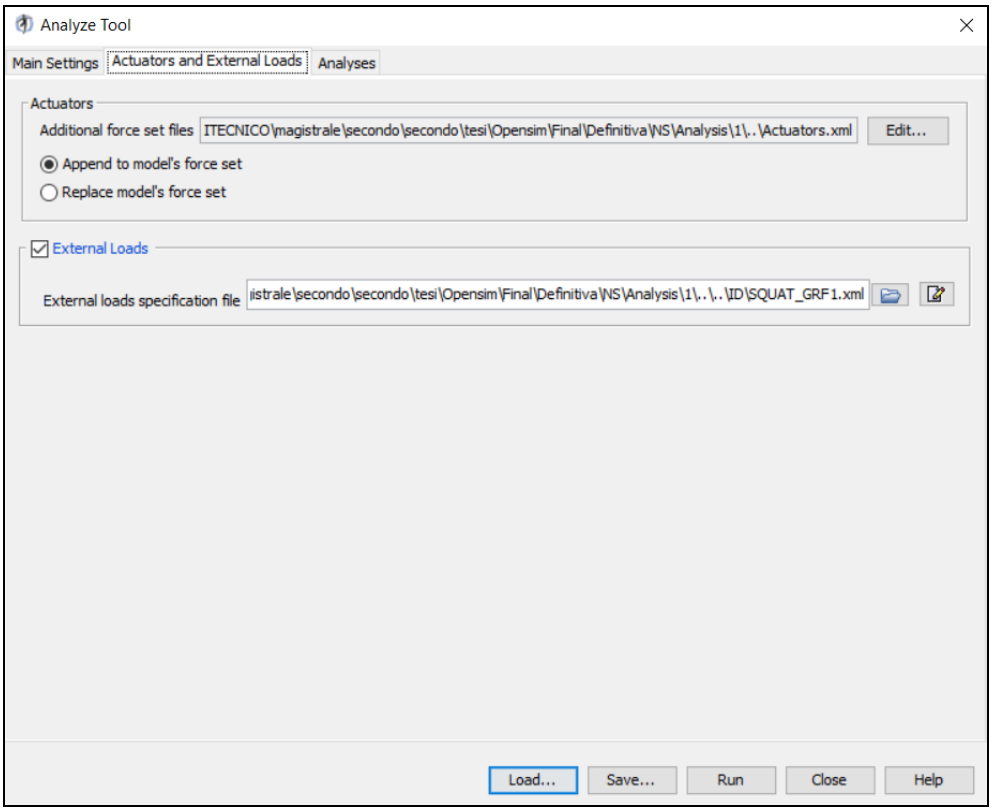


Figure 43: examples of the Analyze Tool interface used in the OpenSim Software**.**

CHAPTER 4
MATERIALS AND METHODS

Let's move on, therefore, to understand more specifically the conceptual path that led to the completion of the whole project. The following subchapter will introduce the protocol and exercises exploited for both simulated (from OpenSim software) and real (from IMU sensors) data.

4.1 Exercises

Each subject performed a different number of repetitions of Normal Squat, Wide Squat and Normal Deadlift. All the data collected were not simply about a correct motion execution, but it was also important to have a collection regarding incorrect exercise (always keeping in mind not to harm permanently or temporarily the subjects recruited). Barbell and weights are related to a load within 50-75% of ISO-MAX in both correct and incorrect configurations.

Incorrectness of exercises executions

NASA Astronaut Strength, Conditioning and Rehabilitation (ASCR) Specialists and the Etic Committee of Politecnico di Milano approved the main execution mistakes here below summarized.

Squat and wide stance squat

1) No straight trunk (figure 44a): implies not maintaining the natural lumbar curves, thus keeping the pelvis in retroversion instead of in anteversion (during squatting) and then rounding back (during rising).
2) Valgus knees (figure 44b): in this case, knees do not follow feet directions, but they approach the medial line of the body.
3) Knee joints over the toes' lines (figure 44c): knees do not remain behind the vertical lines passed through the toes.
4) Heels not in contact with the floor (figure 44c): heels are risen at the end of squatting.
5) Shallow squat: knees flexion angles smaller than 90°.

For simplicity, techniques will be called CO (Correct), RB (Rounded Back), KV (Valgus Knees), KOT (Knees Over Toes), RH (Raised Heels) and SH (SHallow).

Deadlift

1) Rounded back (figure 45a): trunk is not maintained in natural position, but it is bent forward.
2) Bar Over Shoulder (figure 45b): bar is brought too far ahead at the start of exercise, overcoming the shoulders.
3) Hyperextended back (figure 45c): at the end of lifting, lumbar is overextended.

For simplicity, incorrect techniques will be named as RB (Rounded back), BOS (Bar Over Shoulder) and HB (Hyperextended Back) respectively.

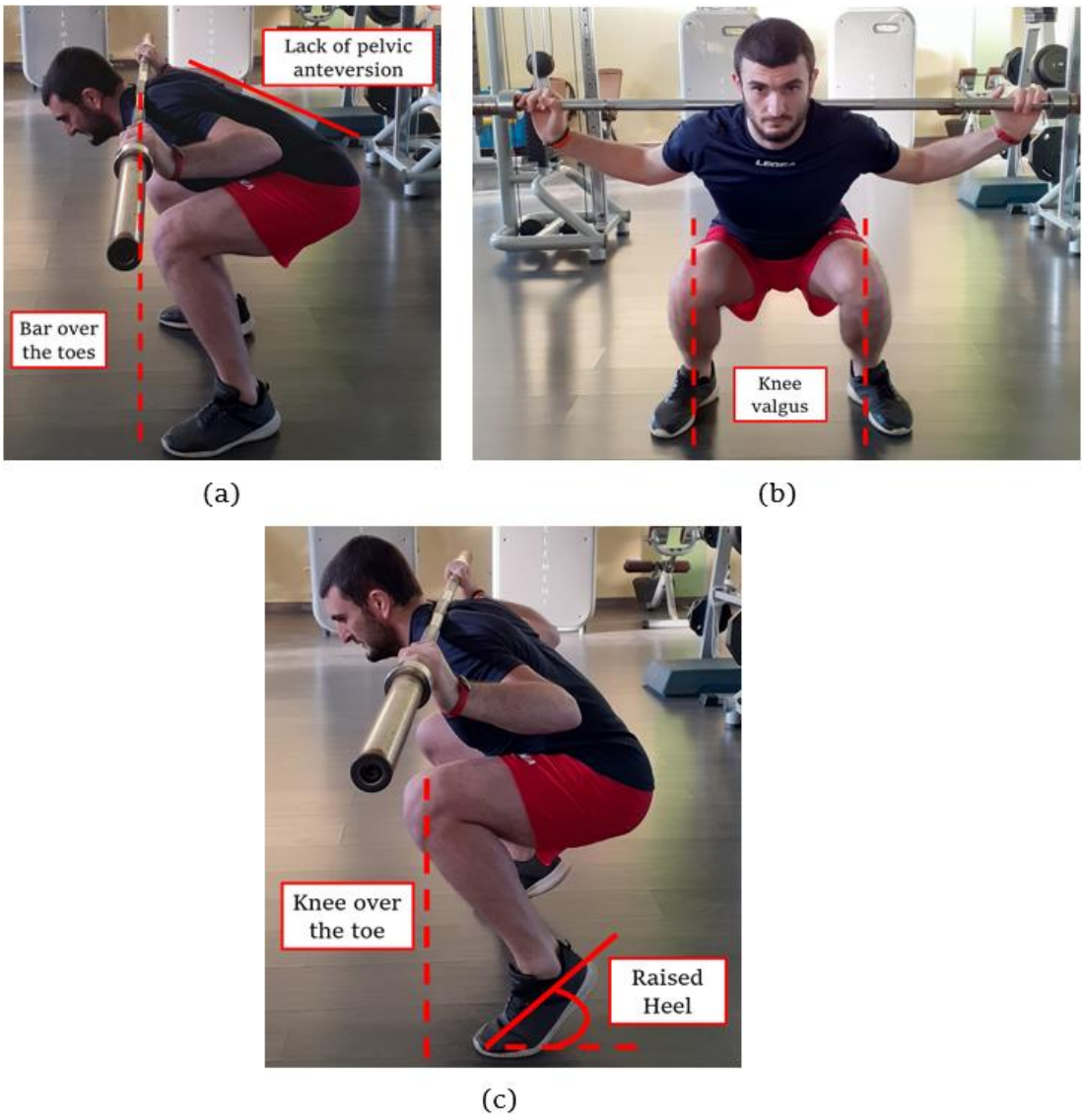


Figure 44: incorrectness of squat. (a) Trunk is not kept straight, and this leads to bring the bar over the vertical lines of toes; dotted line projects shoulder joint on ground highlighting that the line ends in front of the toe. (b) Knees are brought closer; dotted lines show that knees and toes are not aligned. (c) Knees go over toes and heels are raised. Dotted line points out as knee overcomes toe.

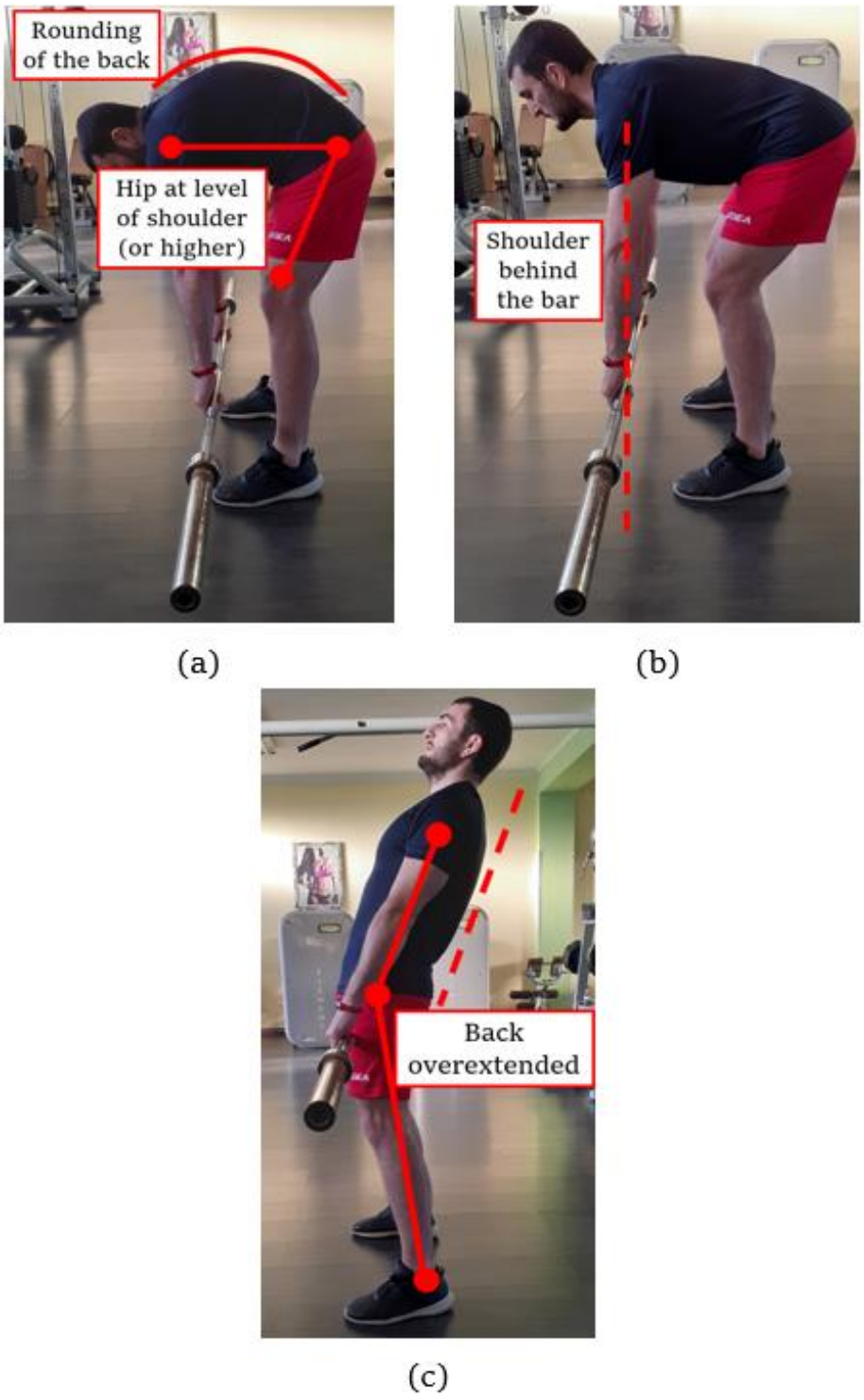


Figure 45: incorrectness of deadlift. (a) Trunk is rounded and this leads to start the exercise with hip and shoulder at the same level instead of having shoulder higher than hip. (b) Dotted line shows the vertical line passing from the shoulder; it is evident that the exercise start with bar in front of that line. (c) At the end of lifting, an excessive lumbar extension is performed, thus shoulder, hip and ankle are not aligned.

4.2 Opensim models used

OpenSim allows to choose among different musculoskeletal models provided by various researchers all over the world. In this project, it was important to have a model that enabled to study and reproduce movements as the Squat exercise. Of course, we are talking about motions characterized by peculiar joint degrees range (e.g., in a Wide Squat, the knee flexion can reach degree values that are even above 140°). Consequently, it was fundamental to choose a musculoskeletal model that fitted these characteristics.

In Literature, the most used and validated model is 'gait2392_simbody', comprising 23-degree-of-freedom of the human body. It features lower extremity joint definitions adopted from Delp et al. (1990), low back joint and anthropometry adopted from Anderson and Pandy (1999), and a planar knee model adopted from Yamaguchi and Zajac (1989). As the name suggests, the number 92 refers to the musculotendon actuators to represent 76 muscles in the lower extremities and torso. The model is a quite good reproduction of the lower extremity body with two legs and torso segment. However, the model itself shows a limitation: it does not contain the upper limbs.

Despite the 'gait2392_simbody' could be considered as a main starting point, this work mainly relied on a musculoskeletal model customized exactly for squatting task and created by Danilo S. Catelli (et Al.). Differently from 'gait2392_simbody' and musculoskeletal models (MSKM) designed to evaluate gait and running, which have limited range of motions (ROMs), the 'Catelli' one granted to provide realistic muscle moment arms (MA) for large ROMs. Indeed, it is suitable for analysis up to 138° hip and 145° knee flexions (the Knee joint is fundamental for exercises like the Squat ones). In addition, it also includes the upper limb body.

Figure 46: screenshot of ‘Catelli’ (left) and ‘gait2392_simbody’ (right) models performing a quat motion.

Even though a scientific research has been already conducted on the Catelli model, we decided to make a ROM comparison with ‘gait2392_simbody’ and verify the superiority (in terms of ROM) of the former.

Considering that the range of motions allowed and computationally used in the ‘gait2392_simbody’ main joints are the following:

- The Hip joint varies from -120° to 120°.
- The Knee joint varies from -10° to 120°.
- The Ankle joint varies from -90° to 90°.

Considering also that the range of motions allowed and computationally used in the ‘Catelli’ main joints are the following:

- The Hip joint varies from -30° to 138°.
- The Knee joint varies from -10° to 145°.
- The Ankle joint varies from -40° to 40°.

To verify the validation, we considered the simulated data coming from 6 subjects recruited at NASA JSC (3 Males, Mean: 32.33 ± 3.06 years old, 79.96 ± 10.84 kg, 174.67 ± 8.50 cm height; 3 Females, Mean: 31.33 ± 6.80 years old, 60.20 ± 7.56 kg, 161.83 ± 7.21 cm height) who performed 4 repetitions for each movement (Normal Squat, Wide Squat and Deadlift) using ARED.
According to the tables and graphs reported in chapter 5.1, we clearly understand and notice the differences between the two models.

4.3 Inertial sensors simulation

Considering the ARED system installed on the ISS, the astronauts have the possibility to collect only kinematics data (acquired with MOCAP system) from their exercises. This statement allows to understand that it is necessary to extract somehow the accelerations data (as the ones obtainable with inertial sensors). In order to reach the purpose afore mentioned, the data (related to the 6 subjects of NASA JSC) already used to verify the validation of "Catelli" musculoskeletal model have also been employed by the OpenSim Analyze tool (please, for any further specification and information about its functioning refer to paragraph 3.4) to simulate the accelerations in a microgravity condition.

The 6 body points chosen (figure 47) are: Sternum, Sacrum, Right Upper Thigh, Left Upper Thigh, Right Upper Shank and Left Upper Shank.

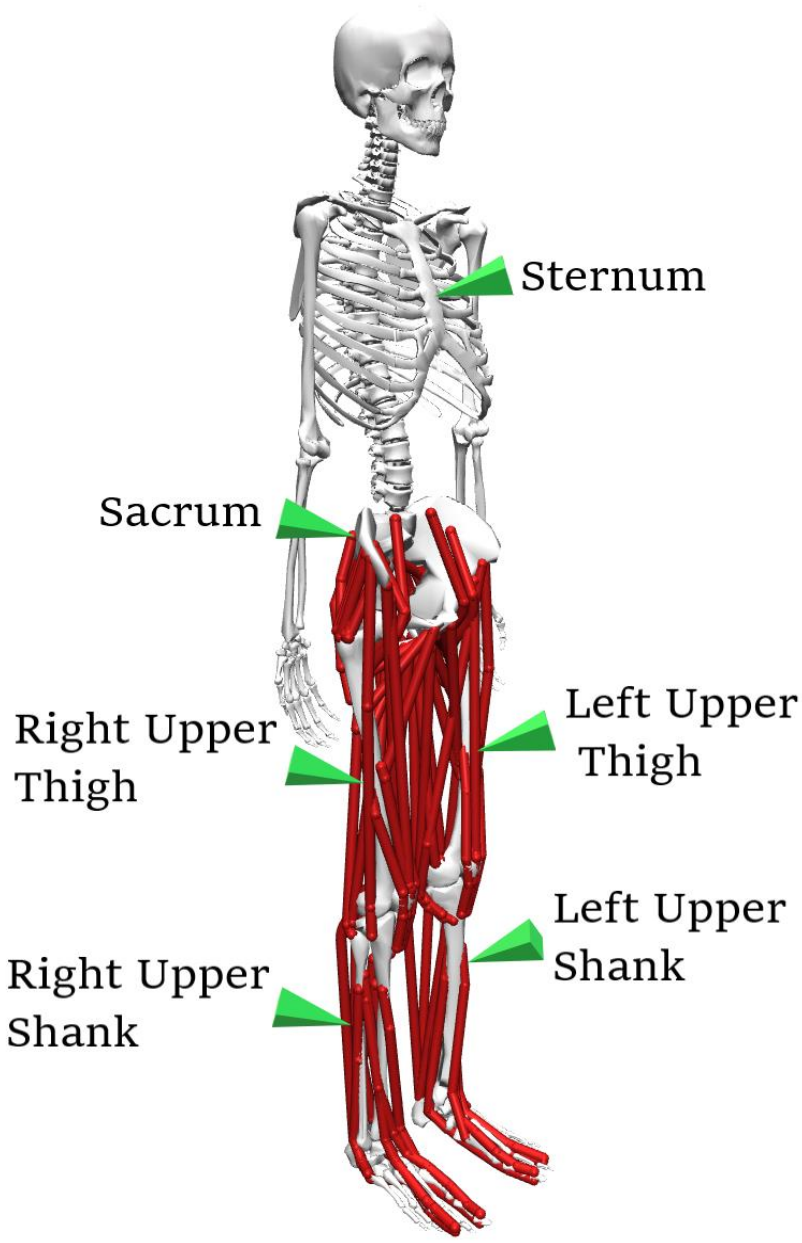


Figure 47: lateral view of the Catelli model with virtual inertial sensors (pointed by the green arrows) placed on Sternum, Sacrum, Right Upper Thigh, Left Upper Thigh, Right Upper Shank and Left Upper Shank.

The OpenSim software provides positions, linear and angular velocities, linear and angular accelerations of each point and saves the result in a .sto file. However, an inexplicable drift of acceleration data was observed, hence they were computed via Matlab deriving only the velocities. An example of the result is shown in figure 48.

Figure 48: accelerations along the three axes of the simulated inertial sensor placed on the Left Upper Thigh.

These data will be useful to understand whether the classification of simulated accelerations while using a classifier trained with real data is a feasible pathway (see chapter 5.5).

4.4 IMU hardware set-up

While the previous subchapters were focused on simulated data, the following topic will treat real ones. Waiting for the dataset collected in-flight by astronauts using the ARED system, we started analyzing all the data collected on-ground by Politecnico di Milano. In particular, data collection was carried out on 17 subjects (9 Males, Mean: 26.89 ± 5.73 years old, 64.22 ± 7.14 kg, 173,44 ± 4.25 cm height; 8 Females, Mean: 25.38 ± 3.77 years old, 56 ± 6.31 kg, 163.14 ± 6.52 cm height) who performed target exercises proposed on ISS (the range varies from 5 to 20, according to their physical capabilities). Acceleration data were acquired during July 2020 in a gym located in Pavia. From the work of Ravizza et al. (2020), no statistical differences are observed between kinematics of exercises performed with ARED and barbell. Consequently, the current thesis project relied on this pillar to conduct the acquisition step.

The Electronic system used (XSENS) to acquire acceleration data is made up of 6 different accelerometers placed in distinct human body locations. In particular, the IMU sensors were placed as follows:

- 1 on sternum.
- 1 on each thigh.
- 1 on each shank.
- 1 on sacrum.

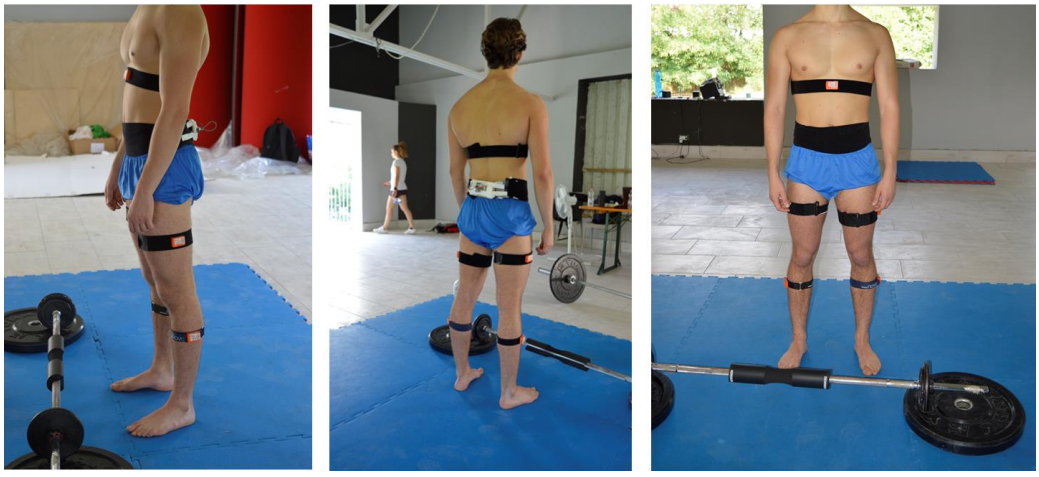


Figure 49: IMU sensors placement for data collection. From left to right, respectively, we have lateral, back and frontal views.

4.5 Data collection and data processing

The raw data used in the current work actually come from another parallel thesis project inserted in the MARS-PRE research. These data were already pre-elaborated by subtracting the gravitational component, in order to simulate the same exact conditions found on ISS.

Basing on the raw information received, our further pre-processing efforts can be summarized according to the following steps:

A) Data **filtering** was related to a sixth-order low pass Butterworth filter, with 5 Hz cut-off frequency.
B) We considered and kept only the **meaningful** acceleration data.
C) The outliers have been **removed**.
D) Every **single repetition** made by the participants was extrapolated, using a peak-detection algorithm applied on the gyroscope signal**.**
E) In order to simplify and accelerate the processing of input data by ML algorithms, all the repetitions have been **resized** according to an average value (e.g., every single repetition lasts **3.5 seconds** and is made up of 350 samples).
F) Dataset **normalization.**
G) **SMOTE** algorithm was applied to solve **imbalance** classification problems (further information is described below in this paragraph).
H) **Feature extraction** and **feature selection** using **PCA** algorithm **(n.b.** applied **only** on the classical ML algorithms and the MLP. The CNN method used acceleration data pre-processed according to the previous steps without considering feature extraction).

Matlab R2019b software was used to execute steps from A to E, while F to H have been carried out through Google Colab (based on python scripts).

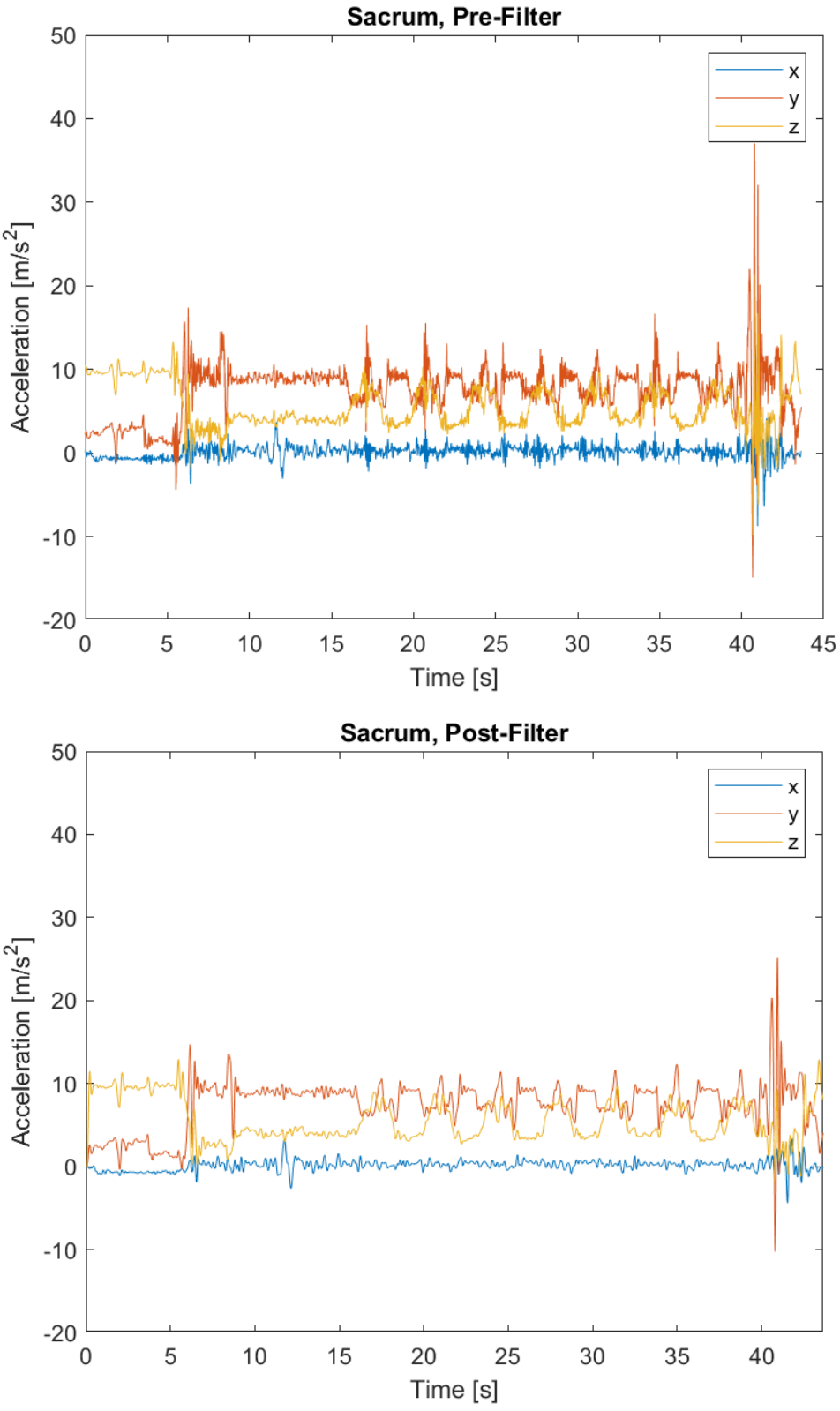


Figure 50: raw acceleration data (x, y, z axis) WITH GRAVITY (just for filter explanation and clarity) coming from the IMU sensor placed on Sacrum (above) and filtered acceleration data (below).

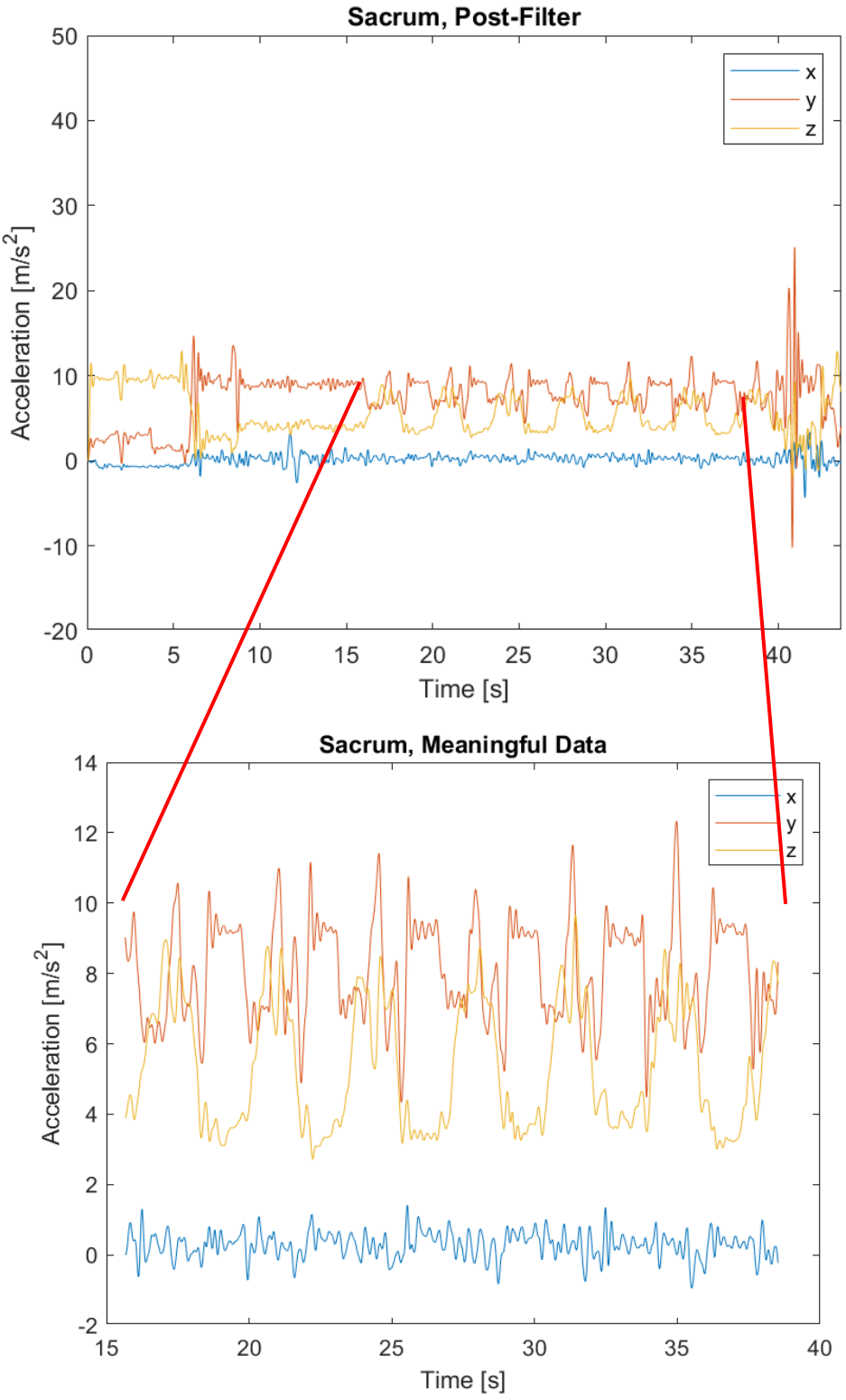


Figure 51: outliers removed; only the meaningful data related to the repetitions of the same exercise have been considered. Initial and final static positions have been cut off. The figure represents, just for explanation clarity, an example of acceleration data WITH GRAVITY.

In order to extract every single repetition made by the participants, a simple peak-detection algorithm was applied on the gyroscope signal with the largest amplitude for any particular exercise. The start and ending points of each repetition can be found by looking for the corresponding zero-crossing points of the gyroscope signal leading up to and following the location of a peak in the signal. Figure 52 demonstrates example results of the segmentation algorithm used on the gyroscope Z signal, from an IMU positioned on the Left Thigh during a Normal Squat exercise.

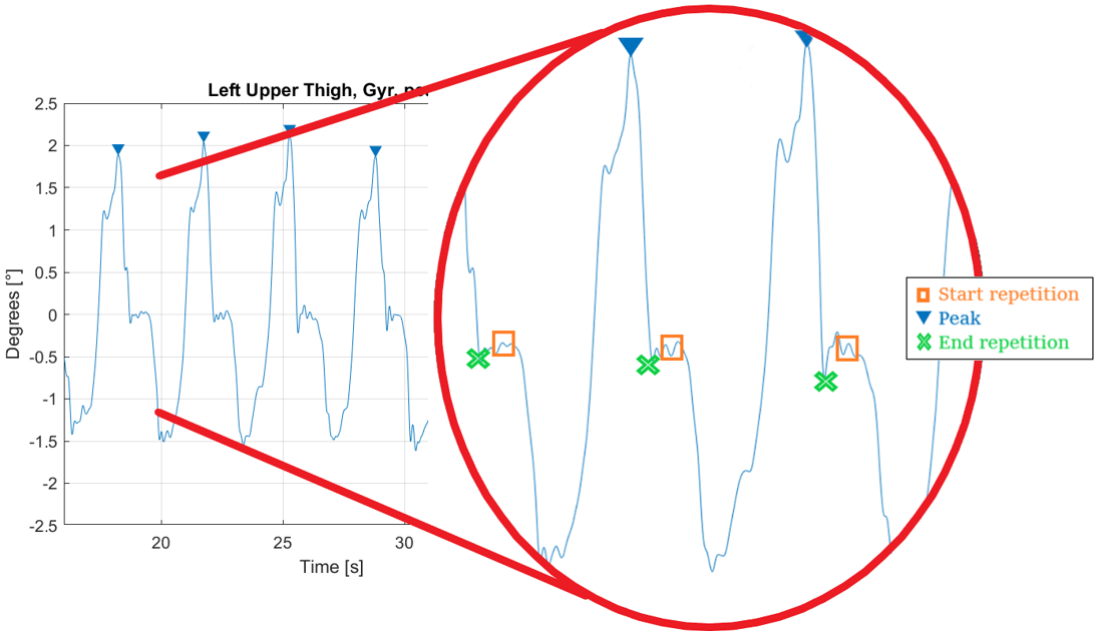


Figure 52: plot showing detection of peak, start, and end points of repetitions through identifying neighboring zero-crossing values to the peak locations. The signal shown is the gyroscope Z signal from the left thigh during a Normal Squat exercise.

4.6 Methods chosen in this work

Even though the acceleration dataset was quite small (only 792, 793 and 508 repetitions were available respectively for Normal Squat, Wide Squat and Deadlift), we chose to consider a multi-label classification problem, in order to distinguish correct and incorrect performances, differentiating the various mistakes.

Six supervised learning methods were compared:

- Support Vector Machine (SVM).

- K-Nearest Neighbor (KNN)
- Decision Tree
- eXtreme Gradient Boosting (XGBoost)
- Multi-layer Perceptron (MLP)
- Convolutional Neural Network (CNN)

Supervised algorithm usually begins with feature extraction. Starting from a set of measured data, non-redundant informative values are extrapolated. These features facilitate the subsequent learning and generalization steps. However, if the dataset extracted is too large and may be redundant, it can be transformed and reduced in a subset by feature selection. The main statistical procedure used for selecting features is Principal Component Analysis (PCA), adopted in this project. The workflow can be observed in the scheme below.

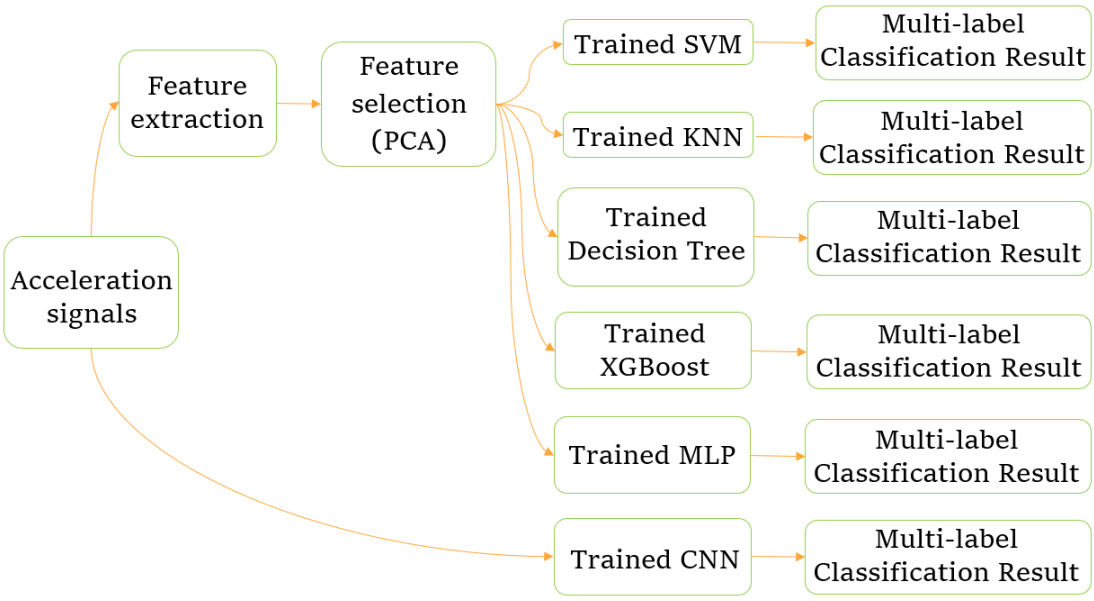


Figure 53: brief schematic explanation of the methods chosen for this work.

The ML algorithms were developed in Google Colab (which is based on python scripts). Training set and test set were randomly created extracting respectively the 70% and 20% of data. A target array was created to define the correct class of each entry, labeled with {0, 1, 2, 3, 4, 5} when considering the Squat exercises (Normal

Squat and Wide Squat), where 0 corresponded to a correct exercise (CO), 1 to Knees Over Toes (KOT), 2 to Valgus Knees (Kv), 3 to Rounded Back (Rb), 4 to Raised Heels (Rh), 5 to Shallow (Sh). When considering Deadlift instead, the labels are {0, 1, 2, 3}, where 0 corresponded to a correct exercise (CO), 1 to Bar Over Shoulder (BOS), 2 to Rounded back (Rb), 3 to Hyperextended back (Hb). Confusion matrix and accuracy of each testing subset were obtained.

Feature extraction

To obtain better results, in terms of accuracy, the current work is based on the extraction of specific features extrapolated from the acceleration data collected (as explained in the previous chapters). However, the feature extraction itself is applied only on the classical machine learning algorithm (e.g. Support Vector Machines SVM, K-Nearest neighbor KNN, Decision Trees, XGBoost) and on the Multi-Layer Perceptron (MLP). As far as the Convolutional Neural Network is concerned, we chose to use, as input, directly the raw data coming from the sensors without considering any kind of feature extraction.

Basing on a GitHub work on the Human Activity Recognition (HAR) classification, we extracted: statistical and geometrical features from raw signals and jerk signals (acceleration first derivative); frequency domain features from raw signals and jerk signals. More precisely, from each sample:

- **x, y and z raw signals**: mean, max, min, standard deviation, skewness, kurtosis, interquartile range, median absolute deviation, area under curve, area under squared curve.
- **x, y and z jerk signals (first derivative)**: mean, max, min, standard deviation, skewness, kurtosis, interquartile range, median absolute deviation, area under curve, area under squared curve.
- **x, y and z raw signals Discrete Fourier Transform**: mean, max, min, standard deviation, skewness, kurtosis, interquartile range, median absolute deviation, area under curve, area under squared curve, weighted mean

frequency, 5 first DFT coefficients, 5 first local maxima of DFT coefficients and their corresponding frequencies.

- **x, y and z jerk signals Discrete Fourier Transform**: mean, max, min, standard deviation, skewness, kurtosis, interquartile range, median absolute deviation, area under curve, area under squared curve, weighted mean frequency, 5 first DFT coefficients, 5 first local maxima of DFT coefficients and their corresponding frequencies.
- **x, y and z correlation coefficients**

Table 8 shows in a concise way a brief list of all the features extracted, together with their related equations.

Table 8: list of all the features extracted; each feature is related to one axis of a single sensor.

Feature	**Formula**
x,y and z raw signals and jerk signals (first derivative)	
Mean	$\bar{x} = \frac{\sum x_i}{N}$
Maximum	max(X)
Minimum	min(X)
Standard Deviation	$\sigma^2 = \sqrt{\frac{\sum (x_i - \bar{x})^2}{N}}$
Skewness	Skewness(X)
Kurtosis	Kurtosis(X)
Interquantile range	Irq(X)
Median Absolute Deviation	MAD=median(|X_i - median(X)|)
Area Under Curve	*(it depends on the data)
Area Under Squared Curve	*(it depends on the data)
x,y and z raw signals and jerk signals DFT	
Mean	$\bar{x} = \frac{\sum x_i}{N}$
Maximum	max(X)
Minimum	min(X)
Standard Deviation	$\sigma^2 = \sqrt{\frac{\sum (x_i - \bar{x})^2}{N}}$
Skewness	Skewness(X)
Kurtosis	Kurtosis(X)
Interquantile range	Irq(X)
Median Absolute Deviation	MAD=median(|X_i - median(X)|)
Area Under Curve	*(it depends on the data)
Area Under Squared Curve	*(it depends on the data)
Weighted Mean Frequency	*
5 first DFT coefficients	$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k e^{\frac{i \pi k n}{N}}$
5 first local maxima of DFT coefficients	max(x_n)
x,y and z correlation coefficients	

Principal Component Analysis

Principal component analysis (PCA) is the most widely known technique of attribute reduction by means of projection. Generally speaking, the purpose of this method is about obtaining a projective transformation that replaces a subset of the original numerical attributes with a lower number of new attributes obtained as their linear combination, without this change causing a loss of information. Experience shows that a transformation of the attributes may lead in many instances to better accuracy in the learning models subsequently developed.

Before applying the principal component method, it is expedient to standardize the data, to obtain for all the attributes the same range of values, usually represented by the interval [−1, 1]. Moreover, the mean of each attribute $\mathbf{a}_j$ is made equal to 0 by applying the transformation

$$\tilde{x}_{ij} = x_{ij} - \frac{1}{m}\sum_{i=1}^{m} x_{ij}.$$

Let **X** denote the matrix resulting from applying the transformation above to the original data and let **V** = **X**'**X** be the covariance matrix of the attributes. If the correlation matrix is used to develop the principal component analysis method instead of the covariance matrix, the transformation is not required.

Starting from the *n* attributes in the original dataset, represented by the matrix **X**, the principal component method derives *n* orthogonal vectors, namely the *principal components*, which constitute a new basis of the space R^n.

Principal components are better suited than the original attributes to explain fluctuations in the data, in the sense that usually a subset consisting of *q* principal components, with $q < n$, has an information content that is almost equivalent to that of the original dataset. As a consequence, the original data are projected into a lower-dimensional space of dimension *q* having the same explanatory capability.

Principal components are generated in sequence by means of an iterative algorithm. The first component is determined by solving an appropriate optimization problem, in order to explain the highest percentage of variation in the data. At each iteration,

the next principal component is selected, among those vectors that are orthogonal to all components already determined, as the one which explains the maximum percentage of variance not yet explained by the previously generated components.
At the end of the procedure the principal components are ranked in non-increasing order with respect to the amount of variance that they are able to explain.

Let $\mathbf{p}_j$, $j \in N$, denote the n principal components, each of them being obtained as a linear combination $\mathbf{p}_j = \mathbf{X}\mathbf{w}_j$ of the available attributes, where the weights $\mathbf{w}_j$ have to be determined. The projection of a generic example $\mathbf{x}_i$ in the direction of the weights vector $\mathbf{w}_j$ is given by $\mathbf{w}'_j\mathbf{x}_i$. It can easily be seen that its variance is given by

$$\begin{aligned} \mathrm{E}[\mathbf{w}'_j\mathbf{x}_i - \mathrm{E}[\mathbf{w}'_j\mathbf{x}_i]]^2 &= \mathrm{E}[(\mathbf{w}'_j(\mathbf{x}_i - \mathrm{E}[\mathbf{x}_i]))^2] \\ &= \mathbf{w}'_j\,\mathrm{E}[(\mathbf{x}_i - \mathrm{E}[\mathbf{x}_i])'(\mathbf{x}_i - \mathrm{E}[\mathbf{x}_i])]\mathbf{w}_j \\ &= \mathbf{w}'_j\mathbf{V}\mathbf{w}_j. \end{aligned}$$

The first principal component $\mathbf{p}_1$ represents a vector in the direction of maximum variance in the space of the original attributes and therefore its weights may be obtained by solving the quadratic constrained maximization problem

$$\max_{\mathbf{w}_1} \quad \{\mathbf{w}'_1\mathbf{V}\mathbf{w}_1 : \ \mathbf{w}'_1\mathbf{w}_1 = 1\},$$

where the unit norm constraint for $\mathbf{w}_1$ is introduced in order to derive a well-posed problem. By introducing the Lagrangian function

$$L(\mathbf{w}_1, \lambda_1) = \mathbf{w}'_1\mathbf{V}\mathbf{w}_1 - \lambda_1(\mathbf{w}'_1\mathbf{w}_1 - 1),$$

and applying the Karush–Kuhn–Tucker conditions, the solution of the maximization problem reduces to the solution of the system

$$\begin{aligned} \frac{\partial L(\mathbf{w}_1, \lambda_1)}{\partial \mathbf{w}_1} &= 2\mathbf{V}\mathbf{w}_1 - 2\lambda_1\mathbf{w}_1 = \mathbf{0}, \\ \frac{\partial L(\mathbf{w}_1, \lambda_1)}{\partial \lambda_1} &= 1 - \mathbf{w}'_1\mathbf{w}_1 = 0, \end{aligned}$$

which can be rewritten as

$$(\mathbf{V} - \lambda_1 \mathbf{I})\mathbf{w}_1 = \mathbf{0},$$

subject to the unit norm condition $\mathbf{w}'_1\mathbf{w}_1 = 1$, where $\mathbf{I}$ is the identity matrix. The solution of the maximization problem is therefore given by $\mathbf{w}_1 = \mathbf{u}_1$, where $\mathbf{u}_1$ is the eigenvector of unit norm associated with the maximum eigenvalue λ_1 of the covariance matrix $\mathbf{V}$. Since the variance we wish to maximize is given by

$$\mathbf{w}_1' \mathbf{V} \mathbf{w}_1 = \lambda_1 \mathbf{w}_1' \mathbf{w}_1 = \lambda_1 \mathbf{u}_1' \mathbf{u}_1 = \lambda_1,$$

the first principal component is obtained by means of the eigenvector $\mathbf{u}_1$, associated with the maximum eigenvalue λ_1 of $\mathbf{V}$, through the relation $\mathbf{p}_j = \mathbf{X}\mathbf{u}_j$.

The second principal component may be determined by solving an optimization problem, adding the condition of orthogonality to the previously obtained principal component, expressed by the constraint

$$\mathbf{w}_2' \mathbf{u}_1 = 0.$$

Proceeding in an iterative way, it is possible to derive the n principal components starting from the eigenvectors $\mathbf{u}_j$, $j \in N$, of $\mathbf{V}$ ordered by non-increasing eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$, through the equalities $\mathbf{p}_j = \mathbf{X}\mathbf{u}_j$. The variance of the principal component $\mathbf{p}_j$ is given by $\text{var}(\mathbf{p}_j) = \lambda_j$.

The n principal components constitute a new basis in the space R^n since the vectors are orthogonal to each other. Therefore, they are also uncorrelated and can be ordered according to a relevance indicator expressed by the corresponding eigenvalue. In particular, the first principal component explains the greatest proportion of variance in the data, the second explains the second greatest proportion of variance, and so on. If $\mathbf{U}$ denotes the $n \times n$ matrix whose columns are the eigenvectors $\mathbf{u}_j$, $j \in N$, and $\mathbf{P}$ indicates the $n \times n$ matrix whose columns are the principal components $\mathbf{p}_j$, $j \in N$, then the equality $\mathbf{P} = \mathbf{X}\mathbf{U}$ holds true.

The total variance of the principal components is equal to the total variance of the original attributes, that is,

$$\sum_{j=1}^{n} \text{var}(\mathbf{p}_j) = \sum_{j=1}^{n} \lambda_j = \text{tr}(\mathbf{V}) = \sum_{j=1}^{n} \text{var}(\mathbf{a}_j).$$

The interpretation of the principal components may be obtained from the coefficients of the vector $\mathbf{w}_j = \mathbf{u}_j$ which express their relationship with the original attributes. To this end, notice that the principal component $\mathbf{p}_h$ assumes the form

$$\mathbf{p}_h = u_{h1}\mathbf{a}_1 + u_{h2}\mathbf{a}_2 + \cdots + u_{hn}\mathbf{a}_n.$$

The coefficient u_{hj} can be therefore interpreted as the weight of the attribute $\mathbf{a}_j$ in determining the component $\mathbf{p}_h$. The greater the absolute value of u_{hj} is, the more the component $\mathbf{p}_h$ is characterized by the attribute $\mathbf{a}_j$. At the same time, var$(\mathbf{p}_h) = \lambda_h$ represents a measure of the proportion of total variance explained by the principal component $\mathbf{p}_h$. For this reason, the index

$$I_q = \frac{\lambda_1 + \lambda_2 + \cdots + \lambda_q}{\lambda_1 + \lambda_2 + \cdots + \lambda_n}$$

expresses the percentage of total variance explained by the first q principal components and provides an indication of the amount of information preserved by the first q components. In order to determine the number of principal components to be appropriately used, it is possible to go on until the level of overall importance I_q of the considered components exceeds a threshold I_{min} deemed reasonable, in relation to the properties of the dataset. The number of principal components is therefore determined as the smallest value q such that $I_q > I_{\text{min}}$.

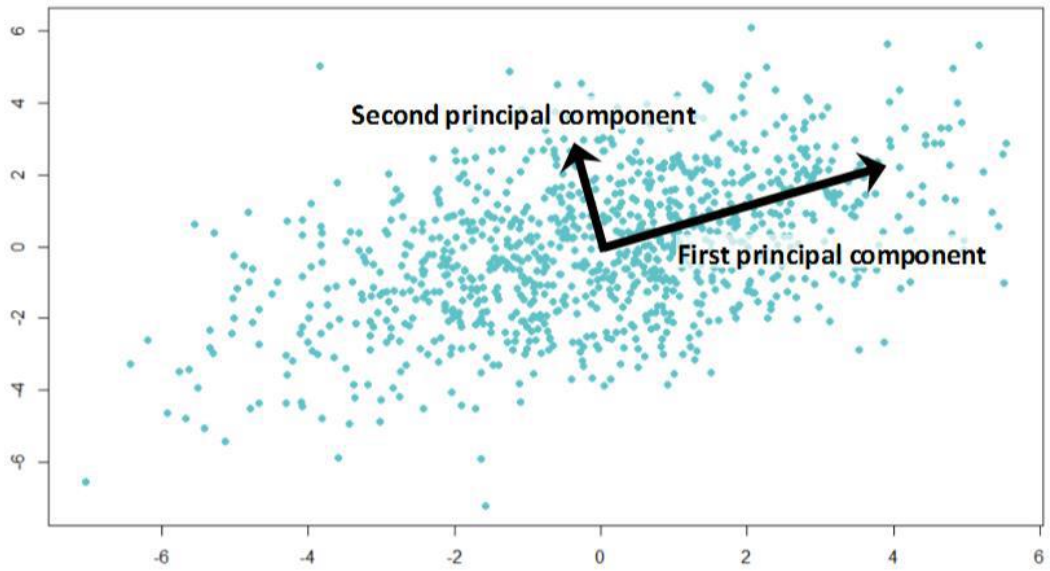


Figure 54: example of the result of PCA in which the first two PCs are plotted; it is clearly visible how the data are mainly distributed along the first Principal component. Moreover, one can also observe how the two PCs are orthogonal.

This work is based on the algorithm developed in Google Colab (based on python script) according to the following steps:

- Standardization of data using z-score $Z = \frac{x_i - \bar{x}}{\sigma_x}$.
- Computation of the covariance matrix V = X′X.
- Calculation of eigenvectors and eigenvalues of V, which represent respectively the directions of the PCs and their amplitudes.
- Choice of the smallest number of PCs able to explain at least the 90% of the total Variance.

Smote algorithm

The challenge of working with imbalanced datasets is that most machine learning techniques will ignore (and, in turn, have poor performance on) the minority class. However, typically, the performance on the minority class happens to be sometimes the most important. As a consequence, the model considered will not effectively learn the decision boundary.

One approach to addressing imbalanced datasets is to oversample the minority class. The simplest approach involves duplicating examples in the minority class: this aim can be achieved by simply duplicating examples from the minority class in the training dataset prior to fitting a model. Even though the result is about balancing the class distribution, these examples do not add any new information to the model. On the other hand, instead, new examples can be synthesized from the existing examples. We are hence considering a specific type of data augmentation for the minority class, better known as Synthetic Minority Oversampling Technique, or SMOTE in short. Perhaps, it is the approach most widely used when synthesizing new examples.

This technique was described by Nitesh Chawla, et al. in their 2002 paper entitled "SMOTE: Synthetic Minority Over-sampling Technique." To have a better understanding of the algorithm, SMOTE works by selecting examples that are close in the feature space, sketching a line between the examples in the feature space and drawing a new sample at a point along that line.

Specifically, a random example from the minority class is first chosen. Then *k* of the nearest neighbors for that example are found. A randomly selected neighbor is chosen, and a synthetic example is created at a randomly selected point between the two examples in feature space.

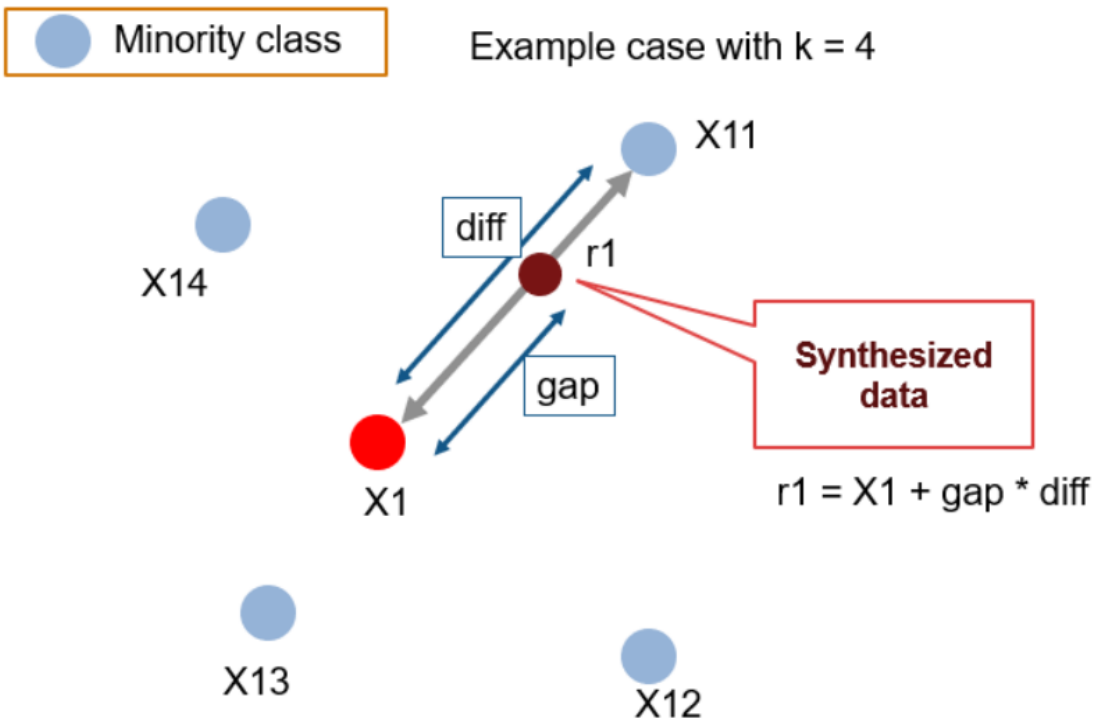


Figure 55: example of the SMOTE algorithm functioning.

This procedure can be used to create as many synthetic examples for the minority class as are required. As described in the paper "Imbalanced Learning: Foundations, Algorithms, and Applications" (2013), the suggestion is about using random under-sampling to trim the number of examples in the majority class, and then using SMOTE to oversample the minority class to balance the class distribution.

To summarize, the approach is effective because new synthetic examples from the minority class are created that are plausible (namely, they are relatively close in feature space to existing examples from the minority class).

A general drawback of the method above explained regards the fact that synthetic examples are created without considering the majority class, possibly resulting in ambiguous examples whether there is a strong overlap for the classes.

Support Vector Machine

Support Vector Machines represent a separation method for classification and estimation. They allow to identify a set of features, called support vectors (SVs), which are representative of the target classes. They are more important than the original features because SVs determine the position of the separation surface obtained from the classifier in the feature space.

In binary classification, SVM create N-dimension hyperplane which optimally separates data into two groups. The separation margin represents the distance between the pairs of canonic hyperplanes, parallel to the separation surface. The points collocated at the minimum distance to the hyperplane of separation are the support vectors (figure 56).

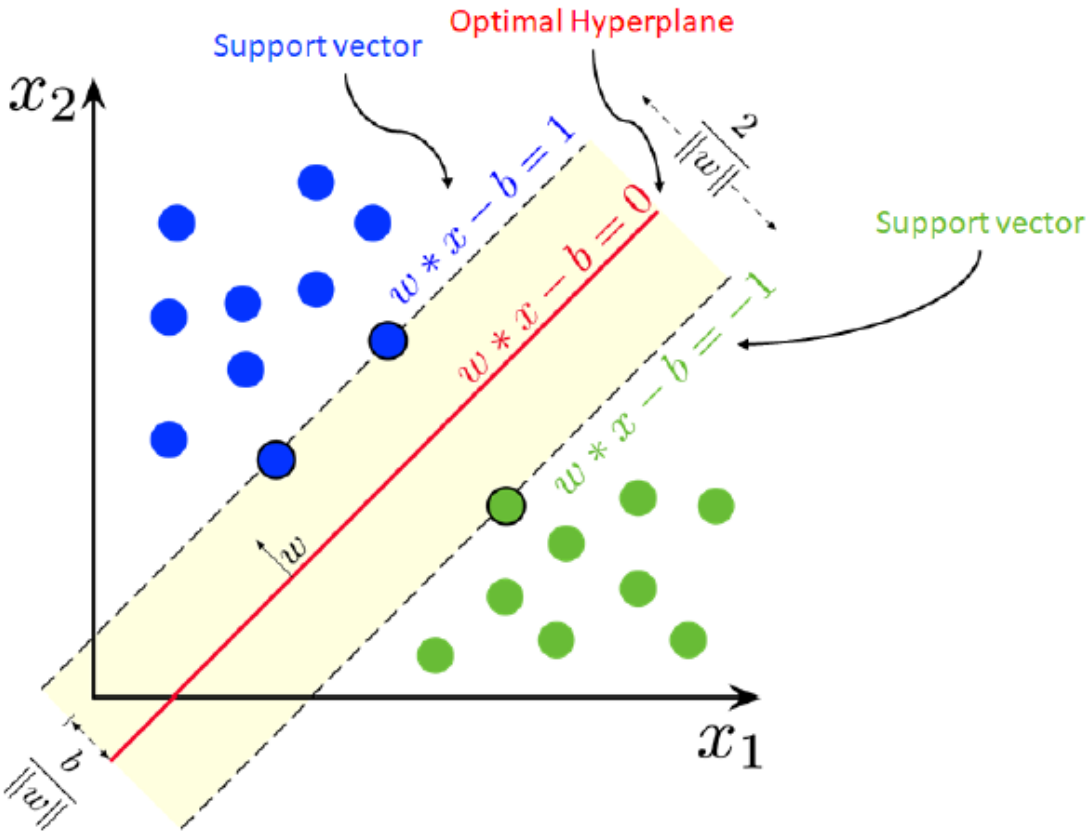


Figure 56: example of the separation surface of the maximum margin obtained with SVM applied for linearly separable data. The optimal hyperplane is shown in red; the maximum margin is represented in yellow; black dotted lines identify the support vectors for linearly separable data, whose formulas are expressed in blue and green. The two colors distinguish data belonging to the two classes.

Calling *w* the angular coefficient of the separator hyperplane and *b* the known term, the equation of the hyperplane is:

$$w'x = b$$

while the equations of the canonic hyperplanes are:

$$w'x - b = 1 \; w'x - b = -1$$

The separation margin δ is:

$$\delta = \frac{2}{\|\omega\|} \qquad \|w\| = \sqrt{\sum_{j \epsilon N} \omega_j^2}$$

The goal of SVM is to maximize the separation margin with constraints, which impose that each point x_i stays in the half-space corresponding to the class y_i. In particular, if data are linearly separable, the optimization problem is:

$$\min_{w,b} \left\{ \frac{1}{2} \|\omega\|^2 \right\} \qquad y_i(wx_i - b) \geq 1$$

Where y_i is the label vector {-1, 1} that determines the class and the optimization problem is seen as the minimization of the reciprocal of the margin.

If data are non-linearly separable, instead, the problem is:

$$\min_{w,b} \left\{ \frac{1}{2} \|\omega\|^2 + \lambda \sum_{i=1}^{m} d_i \right\} \qquad y_i(wx_i - b) \geq 1 - d_i$$

Where $d_i > 0$ is the error of classification; λ adjusts the relative weight of the generalization capability, which is represented by the reciprocal of the margin, and of the accuracy in respect to the training set, expresses as the sum of the errors.

The optimization problem is solved using Lagrange multipliers $\alpha_i \geq 0$ and $\mu \geq 0$ the multiplier of the constraints:

$$L(w, b, d, \alpha, \mu) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{k=1}^{m} y_i y_k \alpha_i \alpha_k x'_i x_k$$

With $\alpha \leq \lambda$.

K-Nearest Neighbors (KNN) algorithm

The k-nearest neighbors' algorithm is a supervised classification algorithm. It takes a bunch of labeled points and uses them to learn how to label other points. To label a new point, the algorithm looks at the already labeled points closest to that new point (that's why we talk about its nearest neighbors) and it will relate each neighbor to a single vote. Hence, this new point will be classified according to a criteria of majority voting. Moreover, the term "k", in K-Nearest Neighbors, refers to the number of neighbors the algorithm checks. It is 'supervised' because we are trying to classify a point basing on the known classification of other points.

KNN works according to the following steps:

- Determine the value for K.
- Calculate the distances between the new input (test data) and all the training data. The most used metrics for calculating distance are Euclidean, Manhattan, Minkowski and Mahalanobis ones.
- Sort the distance and determine k nearest neighbors based on minimum distance values.
- Analyze the category of those neighbors and assign the category for the test data based on majority voting.
- The output of KNN is the predicted class.

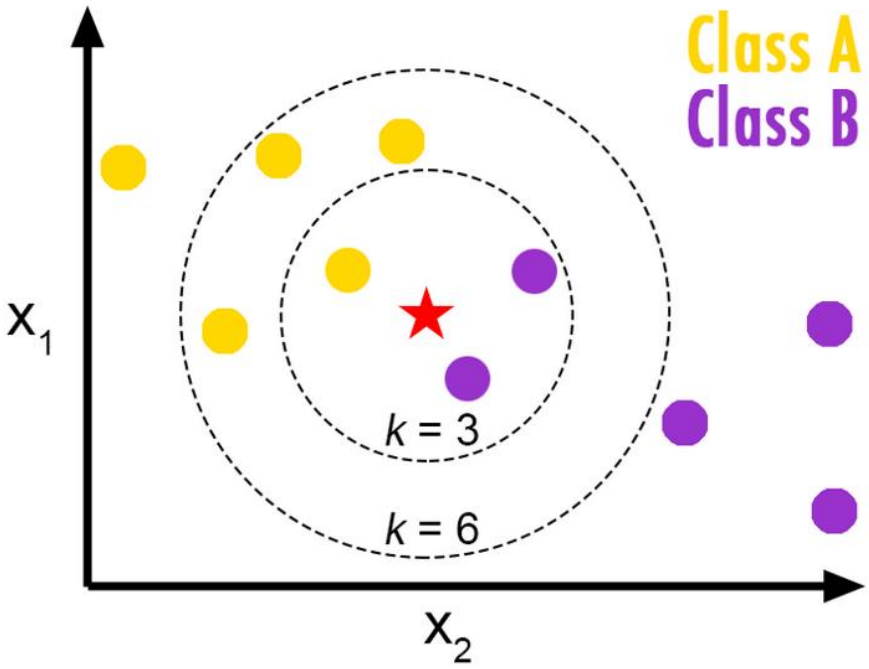


Figure 57: example of KNN. A bunch of already classified data (yellow and violet ones) are used by the algorithm to determine the classification of the new data considered (red star). In this example, if we choose K = 3 the star will join the Class B dataset; considering K = 6, instead, the new point will belong to Class A.

The *Euclidean distance* between the vectors associated with the pair of observations $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ and $\mathbf{x}_k = (x_{k1}, x_{k2}, \ldots, x_{kn})$ in *n*-dimensional space is defined as

$$\begin{aligned} \text{dist}(\mathbf{x}_i, \mathbf{x}_k) &= \sqrt{\sum_{j=1}^{n} (x_{ij} - x_{kj})^2} \\ &= \sqrt{(x_{i1} - x_{k1})^2 + (x_{i2} - x_{k2})^2 + \cdots + (x_{in} - x_{kn})^2}. \end{aligned}$$

As alternative, we may consider the *Manhattan distance*

$$\begin{aligned}\text{dist}(\mathbf{x}_i, \mathbf{x}_k) &= \sum_{j=1}^{n} |x_{ij} - x_{kj}| \\ &= |x_{i1} - x_{k1}| + |x_{i2} - x_{k2}| + \cdots + |x_{in} - x_{kn}|,\end{aligned}$$

which is so called because in order to reach one point from another we have to travel along two sides of a rectangle having the two points as its opposite vertices.

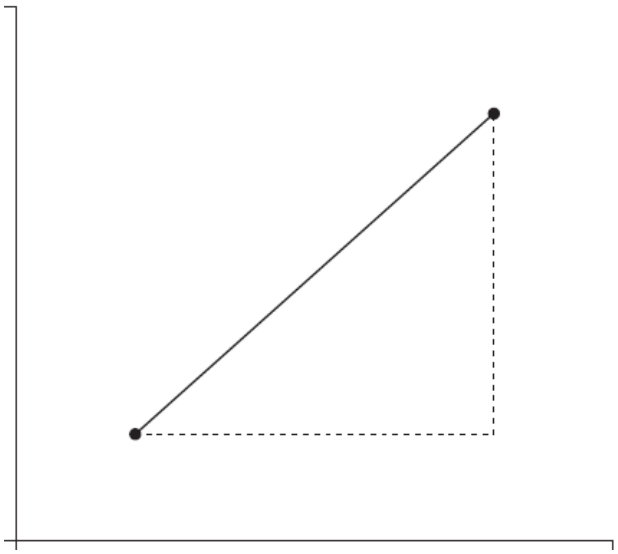


Figure 58: difference between Euclidean (full line) and Manhattan distance (dotted line), between two point in a plane.

A third option, which generalizes both the Euclidean and Manhattan metrics, is the *Minkowski distance*, defined as

$$\begin{aligned}\text{dist}(\mathbf{x}_i, \mathbf{x}_k) &= \sqrt[q]{\sum_{j=1}^{n} |x_{ij} - x_{kj}|^q} \\ &= \sqrt[q]{|x_{i1} - x_{k1}|^q + |x_{i2} - x_{k2}|^q + \ldots, |x_{in} - x_{kn}|^q},\end{aligned}$$

for some positive integer q. The Minkowski distance reduces to the Manhattan distance when $q = 1$, and to the Euclidean distance when $q = 2$.

A further generalization of the Euclidean distance can be obtained through the *Mahalanobis distance*, defined as

$$\text{dist}(\mathbf{x}_i, \mathbf{x}_k) = \sqrt{(\mathbf{x}_i - \mathbf{x}_k)\mathbf{V}_{ik}^{-1}(\mathbf{x}_i - \mathbf{x}_k)'},$$

where $\mathbf{V}^{-1}{}_{ik}$ is the inverse of the covariance matrix of the pair of observations $\mathbf{x}_i$ and $\mathbf{x}_k$. If the observations $\mathbf{x}_i$ and $\mathbf{x}_k$ are independent, so that the covariance matrix reduces to the identity matrix, the Mahalanobis distance coincides with the Euclidean distance. The Mahalanobis distance is used when the observations are highly correlated, with different variances and a different range.

Classification trees

Classification trees are perhaps the best-known and most widely used learning methods in data mining applications. The reasons for their popularity lie in their conceptual simplicity, ease of usage, computational speed, robustness with respect to missing data and outliers and, most of all, the interpretability of the rules they generate. To separate the observations belonging to different classes, methods based on trees obtain simple and explanatory rules for the relationship existing between the target variables and predictive ones.

The development of a classification tree corresponds to the training phase of the model and is regulated by a recursive procedure of heuristic nature, based on a divide-and-conquer partitioning scheme.

The steps referred to the classification methods used by decision trees are the following:

1. In the initialization phase, each observation is placed in the root node of the tree. The root is included in the list *L* of active nodes.
2. If the list *L* is empty the procedure is stopped, otherwise a node *J* belonging to the list *L* is selected, removed from the list and used as the node for analysis.

3. The optimal rule to split the observations contained in *J* is then determined, based on an appropriate preset criterion. The splitting rule generated in this way is then applied, and descendant nodes are constructed by subdividing the observations contained in *J*. For each descendant node, the conditions for stopping the subdivision are verified. If these are met, node *J* becomes a leaf, to which the target class is assigned according to the majority of the observations contained in *J*. Otherwise, the descendant nodes are added to the list *L*. Finally, step 2 is repeated.

Splitting rules: for each node of the tree, it is necessary to specify the criteria used to identify the optimal rule for splitting the observations and for creating the descendant nodes. Several alternative criteria can be considered and they differ in the number of descendants, the number of attributes and the evaluation metrics.

Stopping criteria: at each node of the tree different *stopping* criteria are applied to establish whether the development should be continued recursively, or the node should be considered as a leaf. Even in this case, various criteria have been proposed, which result in quite different topologies of the generated trees, all other elements being equal.

Pruning criteria: finally, it is appropriate to apply a few *pruning* criteria, first to avoid excessive growth of the tree during the development phase (*pre-pruning*), and then to reduce the number of nodes after the tree has been generated (*post-pruning*).

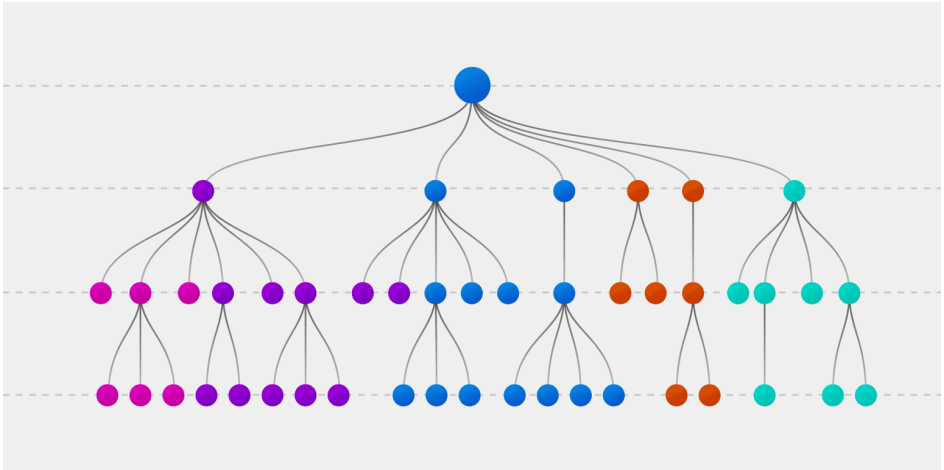


Figure 59: example of Decision Tree structure.

eXtreme Gradient Boosting (XGBoost)

XGBoost can be considered one of the most popular algorithms in the world of data science. Its multitasking aspect allows it to be used in regression or classification projects. It can be used on tabular, structured, and unstructured data.

XGBoost, also named eXtreme Gradient Boosting, refers to a based-tree algorithm (Chen and Guestrin, 2016). It is, hence, part of the tree family (Decision tree, Random Forest, bagging, boosting, gradient boosting).
Boosting is an ensemble method with the primary objective of reducing bias and variance. The goal is about creating weak trees sequentially so that each new tree (or learner) focuses on the weakness (misclassified data) of the previous one. Once a weak learner is added, the data weights are readjusted (a phenomenon generally known as "re-weighting"). The whole forming a strong model after convergence due to the auto-correction once every new learner is added.

The strength of XGBoost is based on parallelism and hardware optimization. The data is stored in in-memory (called a block) and stored in the compressed column [CSC] format. The algorithm can perform tree pruning in order to remove branches with a low probability. The loss function of the model has a term to penalize the complexity of the model with regularization to smooth the learning process (thus decreasing the possibility of overfitting).

The model performs well even with missing or lots of zero values with sparsity awareness. XGBoost uses an algorithm called "weighted quantile sketch algorithm", which allows the algorithm itself to focus on data that are misclassified. The goal of each new learner is simply to learn how to classify the wrong data after each iteration. The method allows you to sort the data by quantiles in order to find the right splitting point. It is actually the goal of the ϵ parameter, which refers specifically to the value of split (ϵ=0.1; quantiles = [10%, 20%, …, 90%]).

The number of iterations for the boosting process is automatically determined by the algorithm with an integrated cross-validation method.
The authors provide a table (table 9) with a comparison of different tree algorithms.

Table 9: comparison of the different tree algorithms according to the studies and works headed by Chen and Guestin (2016. p.7, Table 1).

Table 1: Comparison of major tree boosting systems.

System	exact greedy	approximate global	approximate local	out-of-core	sparsity aware	parallel
XGBoost	yes	yes	yes	yes	yes	yes
pGBRT	no	no	yes	no	no	yes
Spark MLLib	no	yes	no	no	partially	yes
H2O	no	yes	no	no	partially	yes
scikit-learn	yes	no	no	no	no	no
R GBM	yes	no	no	no	partially	no

Feed-forward Artificial Neural Network (MLP, Multi-layer perceptron)

In order to have a clear understanding of what a MLP actually is, we need to introduce the mathematical meaning of a single Rosenblatt perceptron.

The *perceptron*, shown in figure 60, is the simplest form of neural network and corresponds to a single neuron that receives as input the values $(x_1, x_2, \ldots, x_n)$ along the incoming connections, and returns an output value $f(x)$. The input values coincide with the values of the explanatory attributes, while the output value determines the prediction of the response variable y. Each of the n input connections is associated with a weight w_j. An activation function g and a constant ϑ, called *distortion*, are also assigned. Suppose that the values of the weights and the distortion have already been determined during the training phase. The prediction for a new observation **x** is then derived by performing the following steps.

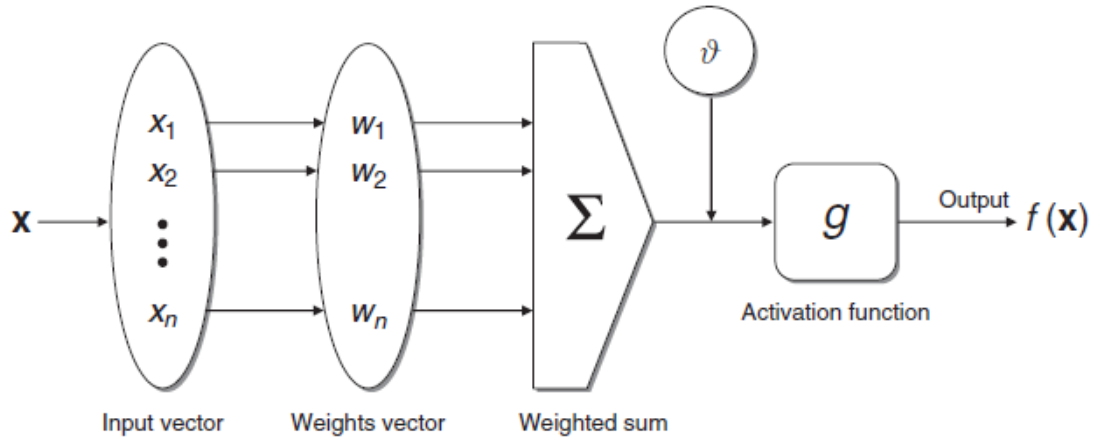


Figure 60: operation of a single unit in a neural network.

First, the weighted linear combination of the values of the explanatory variables for the new observation is calculated and the distortion is subtracted from it:

$$w_1x_1 + w_2x_2 + \cdots + w_nx_n - \vartheta = \mathbf{w'x} - \vartheta.$$

The prediction $f(\mathrm{x})$ is then obtained by applying the activation function g to the linear combination of the predictors:

$$f(\mathrm{x}) = g(w_1x_1 + w_2x_2 + \cdots + w_nx_n - \vartheta) = g(\mathbf{wx} - \vartheta).$$

The purpose of the function g is to map the linear combination into the set $H = \{v_1, v_2, \ldots, v_H\}$ of the values assumed by the target variable, usually by means of a sigmoid profile. For binary classification problems we have $H = \{-1, 1\}$, so that one may select $g(\cdot) = \mathrm{sgn}(\cdot)$, making the prediction coincide with the sign of the weighted sum (in the first equation of this section).

$$f(\mathrm{x}) = \mathrm{sgn}(w_1x_1 + w_2x_2 + \cdots + w_nx_n - \vartheta) = g(\mathbf{w'x} - \vartheta).$$

An iterative algorithm is then used to determine the values of the weights w_j and the distortion ϑ, examining the examples in sequence, one after the other. For each example x_i the prediction $f(\mathrm{x}_i)$ is calculated, and the value of the parameters is then updated using recursive formulas that take into account the error $y_i - f(\mathrm{x}_i)$.

For binary classification problems it is possible to give a geometrical interpretation of the prediction obtained using a Rosenblatt perceptron. Indeed, if we place the m observations of the training dataset in the space R^n, the weighted linear combination calculated for x*i* expresses the slack between the observation and the hyperplane:

$$z = w_1x_1 + w_2x_2 + \cdots + w_nx_n - \vartheta = \boldsymbol{wx} - \vartheta.$$

The purpose of the activation function $g(\cdot) = \text{sgn}(\cdot)$ is therefore to establish if the point associated with the example x_i is placed in the lower or upper half-space with respect to the separating hyperplane. Hence, the Rosenblatt perceptron corresponds to a linear separation of the observations based on the target class. The aim of the iterative procedure is therefore to determine the coefficients of the separating hyperplane.

A *multi-layer feed-forward* neural network (MLP), shown in figure 61, is a more complex structure than the perceptron since it includes the following components.

Input nodes: The purpose of the input nodes is to receive as input the values of the explanatory attributes for each observation. Usually, the number of input nodes equals the number of explanatory variables.

Hidden nodes: Hidden nodes apply given transformations to the input values inside the network. Each node is connected to incoming arcs that go from other hidden nodes or from input nodes, and it is connected with outgoing arcs to output nodes or to other hidden nodes.

Output nodes: Output nodes receive connections from hidden nodes or from input nodes and return an output value that corresponds to the prediction of the response variable. In classification problems, there is usually only one output node.

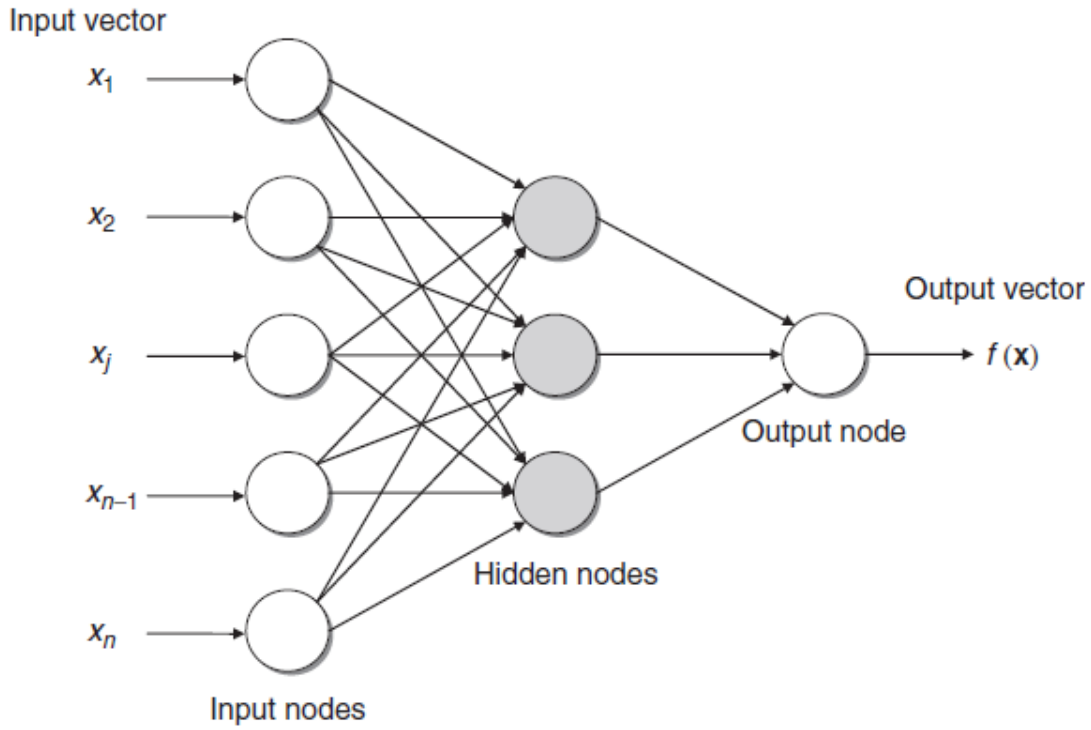


Figure 61: example of MLP. We can clearly observe the presence of input vectors, input nodes, hidden nodes and output nodes.

Each node of the network basically operates as a perceptron, in the sense that given weights are associated with the input arcs, while each node is associated with a distortion coefficient and an activation function. In general, the activation function may assume forms that are more complex than the sign function sgn*(·)*, such as a linear function, a sigmoid or a hyperbolic tangent.

One of the strengths of neural network is about being a learning mechanism applicable to both classification and regression problems. Considering this thesis project, the classification problem is taken into account. More specifically, the two layers feed-forward neural network was developed using Google Colab and the libraries associated with ANNs and MLPs. The Input layer was composed by a number of nodes depending on the result of PCA, which varies for each exercise (Normal Squat, Wide Squat, Deadlift). The number of hidden neurons was selected basing on the minimum error on validation sets. The output layer needed 6 neurons

to classify correct and incorrect exercises (in the same way as SVMs. See previous subchapter for further information). The activation function chosen was the ReLu (Rectified Linear Activation Function).

Convolutional Neural Networks (CNN)

CNNs belong to the category of deep learning methods. They are worldwide known and used in images application. However, in recent year, this algorithm has been applied even in Human activity recognition for both real time and offline situations. As far as this thesis project is concerned, we used the main concepts to elaborate and create a new neural network capable of training on all the sensors used to acquire acceleration data in order to provide information on correct or incorrect exercise execution. For completion, the following sections describe both the CNNs' concepts applied on images and the structure of our CNN created ad-hoc for Human Activity Recognition (HAR).

CNN for images

Convolutional neural networks (CNN) are FFNN designed to exploit spatially-local correlation by enforcing a local connectivity pattern between neurons of adjacent layers. In other words, the inputs of hidden units in layer m are from a subset of units in layer m-1, units that have spatially contiguous receptive fields. This wiring modality of the CNN differs therefore from the traditional full connection related to the Feed forward neural network (FFNN). This paradigm is called sparse connectivity (see figure 62).

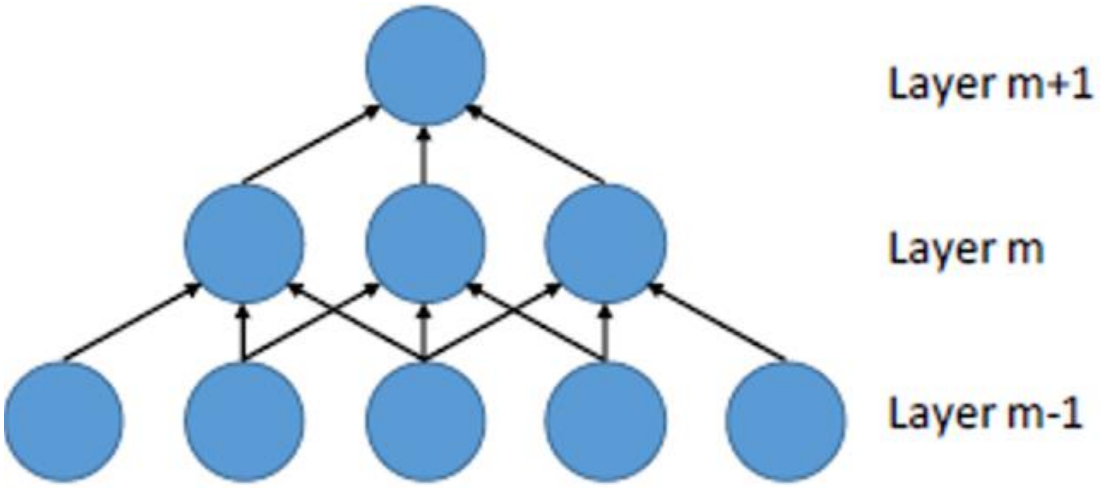


Figure 62: example of CNN wiring exploiting spatially-local correlation.

In figure 62, units in layer m have receptive fields of width 3 in the input and are thus only connected to 3 adjacent neurons in the first layer. Units in layer m+1 have a similar connectivity with the layer below. Therefore, their receptive field with respect to the layer below is also 3, but their receptive field with respect to the input is larger (precisely, 5). Each unit is unresponsive to signals outside of its receptive field with respect to the first layer. The architecture thus ensures that the learnt "filters" produce the strongest response to a spatially local input pattern. However, stacking many such layers leads to (non-linear) "filters" that become increasingly "global" (i.e., responsive to a larger region of pixel space). For example, the unit in hidden layer m+1 can encode a non-linear feature of width 5 (in terms of pixel space).

Convolutional neural networks use three basic ideas: *local receptive fields*, *shared weights*, and *pooling*. A local receptive field consists of a set of connections (which determines the local connectivity), along the corresponding weight set (which determines the filter property), that is replicated across the entire visual field. These replicated units share the same parameterization (weights and bias) and form a feature map. Replicating units in this way allows for features to be detected regardless of their position in the visual field. Additionally, weight sharing increases learning efficiency by greatly reducing the number of free parameters being learnt

(figure 63). The constraints on the model enable CNNs to achieve better generalization on vision problems.

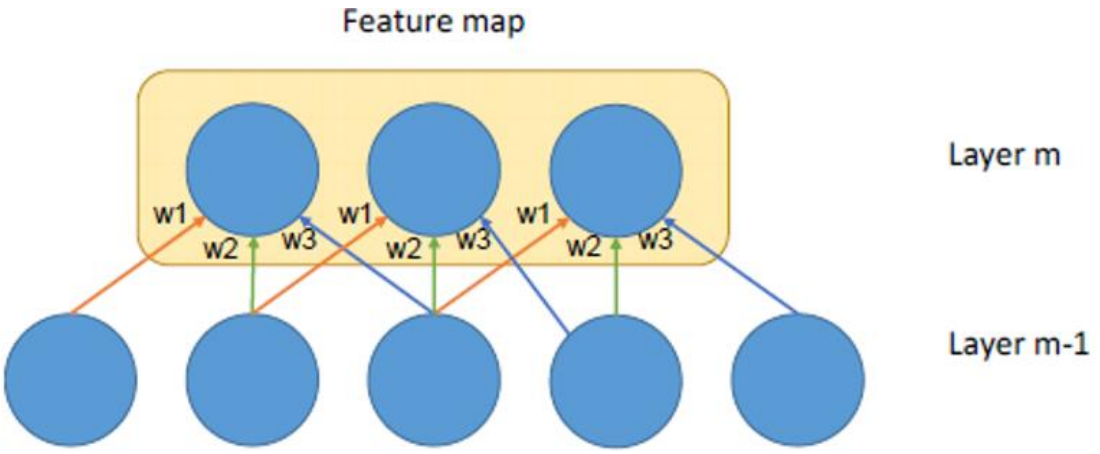


Figure 63: three hidden units belonging to the same feature map. Weights of the same color are shared – constrained to be identical.

A feature map is attained by repetitive application of a function across sub-regions of the whole image, in other words, by *convolution* of the input image with a linear filter, adding a bias term and then applying the neural activation function. If we designate/call the *i*-th feature map at a given layer as u(i), whose filter is determined by the weight matrix W(i) and bias b(i), then the feature map u(i) at the specific location (j, k) in the layer reads as follows:

$$u_{j,k}^{(i)} = tanh\left(b^{(i)} + \sum_{l=1}^{f_x} \sum_{m=1}^{f_y} w_{l,m}^{(i)} a_{j+l,k+m} \right)$$

where f_x and f_y are the horizontal and vertical sizes of the filter, and $a_{x,y}$ is the input activation at position *x,y*. When considering the first convolutional layer then input activation corresponds to the image pixel value. By convention, the bias b is equivalent to the traditional threshold multiplied by the virtual input -1. For example, provided that we have a 40x40 pixel image and we set a local receptive field of 5x5, the output of the first neuron mapping the receptive field will be computed as:

$$u_{1,1} = tanh\left(b + \sum_{l=1}^{5}\sum_{m=1}^{5} w_{l,m} a_{1+l,1+m}\right)$$

requiring only 26 parameters (25 weights + 1 threshold).

Assuming that we adopt a zero-padding strategy (adding zeros to the image boundary) for the convolutional product, the output size (s_s, s_y) of each convolutional layer can be computed as:

$$s_x = \frac{(I_x + 2z_x - f_x)}{L_x} + 1$$

$$s_y = \frac{(I_y + 2z_y - f_y)}{L_y} + 1$$

where I_x and I_y are the input image sizes in x and y directions, z_x and z_y are the zero-padding sizes, f_x and f_y are the filtering sizes, and L_x and L_y are the stride lengths of the filter. The stride accounts for quantification (in pixel) of the shift the filter is undergoing when convoluted with the image. A stride length of 1 will correspond to the filter shift of 1 pixel. A stride length of 3 will correspond to the filter shift of one 3 pixels. Let us assume a 240x240 pixel images, a convolutional filter of 7x7 pixels with a stride length of 3 and a 2-pixel zero padding both in horizontal and vertical directions. The output of the convolution will be then an image map of 80x80 pixels.

An important peculiar characteristic of CNN to keep in mind is that the network is structurally NOT fully connected.

This ML algorithm in general is applied both on 2D images and 3D images. In the latter situation, the images themselves are described in terms of number of pixels in x and y direction, along with the number of slices taken in the z direction. Applying a convolutional filter to such data means to define not only the x, y size but even the z one which specifies how many slices will be processed simultaneously. Likewise, in

this *3D convolution* that is determined by the corresponding *3D feature map*, we can compute the output size of the convolutional processing as follows:

$$s_x = \frac{(I_x + 2z_x - f_x)}{L_x} + 1$$
$$s_y = \frac{(I_y + 2z_y - f_y)}{L_y} + 1$$
$$s_z = \frac{(n_s + 2z_z - f_z)}{L_z} + 1$$

where n_s is the number of slices of the scan, f_z is the number of slices addressed by the filter, L_z is the stride of the filter in the *z* direction. Considering an input volume of 65x65x8, composed say by 33800 voxels, a 5x5x2 filter with no padding and stride equal to 2 in all the three direction, the feature map will endow 3844 neurons, all featuring 51 parameters (50 weights and 1 threshold). This feature map can be regarded as a 3D distribution of neurons regularly distributed on a 3D grid of 31x31x4. Assuming that the first convolutional layer encompasses *n* different features maps, one neuron in the second layer belonging to the first feature map will be spanning all the feature maps in the previous layer. Provided that the filter size is 3x3x2 with no padding and stride of 2 along all the three directions, then the complete feature map will consist of 15x15x2 neurons, i.e. 450 neurons in total. Each neuron therefore will have '(3x3x2) x n + 1' free parameters.

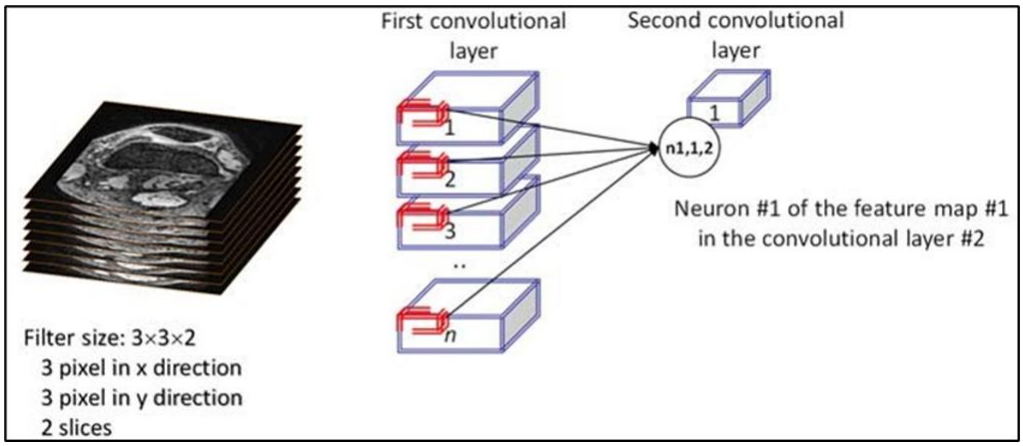


Figure 64: considering a filter size of 3x3x2 with no padding and stride of 2 along all the three directions, the complete feature map will consist of 15x15x2 neurons, i.e. 450 neurons in total. Each neuron therefore will have '(3x3x2) x n +1' free parameters.

Another important concept of CNNs is *max-pooling*, which is a form of non-linear down- sampling. Max-pooling partitions the input image into a set of non-overlapping rectangles and, for each such sub-region, outputs the maximum value. Max-pooling is useful mainly for two reasons: 1) by eliminating non-maximal values, it reduces computation for upper layers; 2) it provides a form of translation invariance.

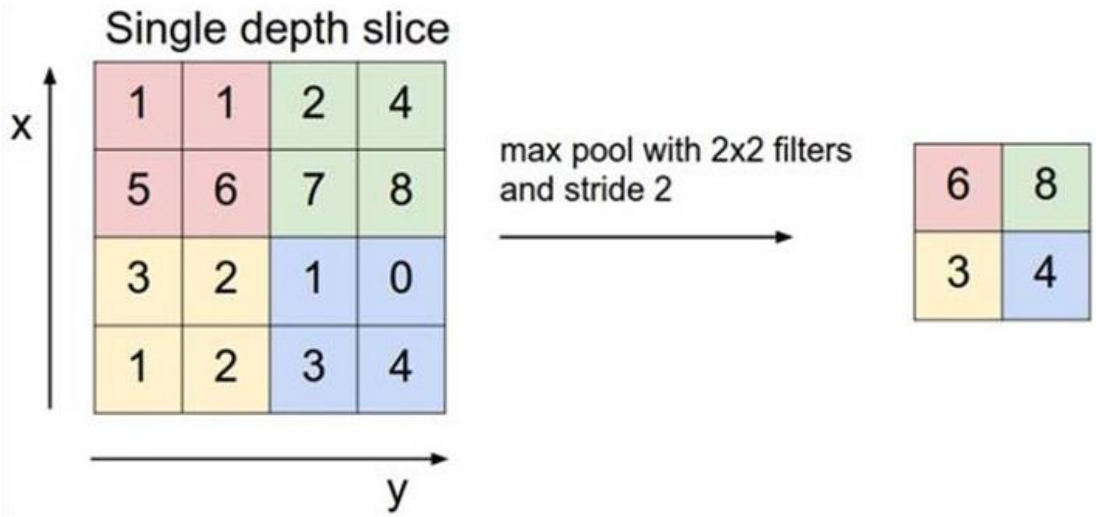


Figure 65: max pooling example considering 2x2 filters and stride 2.

Imagine cascading a max-pooling layer with a convolutional layer (figure 65). There are 8 directions in which one can translate the input image by a single pixel. If max-pooling is done over a 2x2 region, 3 out of these 8 possible configurations will produce exactly the same output at the convolutional layer. For max-pooling over a 3x3 window, this jumps to 5/8. Since it provides additional robustness to position, max-pooling is a "smart" way of reducing the dimensionality of intermediate representations.

Another processing layer utilized in CCN is represented by the rectified linear unit layer (ReLU). It is chained to the convolutional layer to ensure that all the feature maps have positive signals (remember that *activation function tanh* is associated to *potential negative outputs*).

Figure 66: equations and graphical representation of ReLU.

The basic idea of building deep learning with CNN leads to bind together blocks constituted by the three elements above (Convolutional layer, ReLU, Max Pooling) with decreasing number of units up to a final stage of feature classification. Usually this consists of a fully connected FFNN shaped as a cluster network (e.g. softmax) whose multi-dimensional output refers to the number of predefined features or classes (*Fig. 73*). Softmax network maps a N-dimensional vector of arbitrary real values to a N-dimensional vector of real values in the range (0, 1) that add up to 1. This is obtained using normalized exponential activation as: $z_i = eP_i / \sum eP_j$. The CNN can be trained using the traditional algorithm of backpropagation. ReLU and Pooling layers do not require learning.

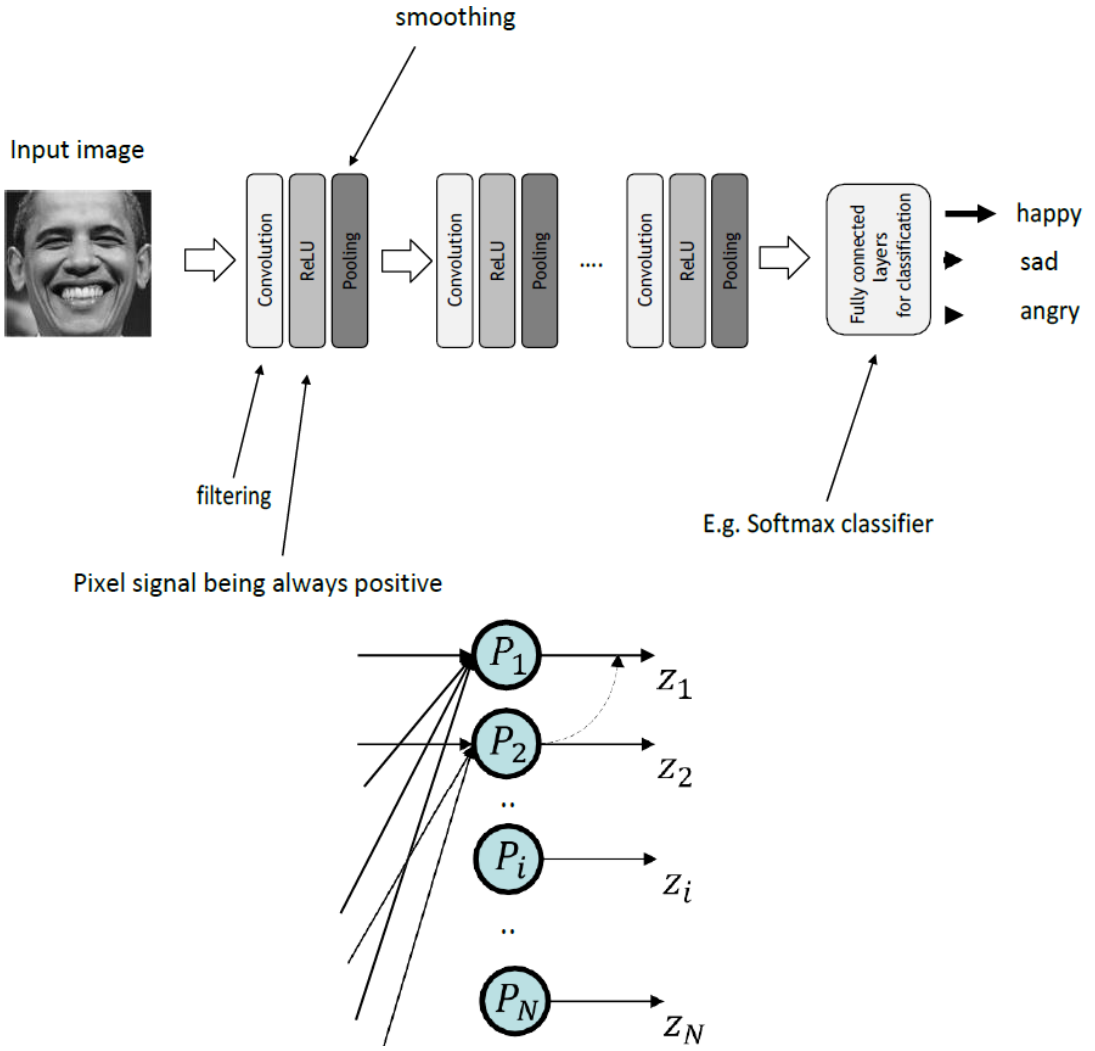


Figure 67: (Upper panel) Example of a classifier built using multi-layer convolutional neural network. The last layer consists of a fully connected FFNN shaped as a cluster network (e.g., softmax). (Lower panel) Softmax connectivity. It “maps" a N-dimensional vector of arbitrary real values to a N-dimensional vector of real values in the range (0, 1) that add up to 1. This is obtained using normalized exponential activation as: $z_i = eP_i / \sum eP_j$.

CNN for real time and offline classification

As previously mentioned, CNNs can be employed not only for image classification but also used in many other applications related to our daily life.

In the current work, the Convolutional Neural Network is applied on the data collected by IMU sensors during Squat and Deadlift exercises to classify correct and

incorrect motions. In order to achieve these goals, we chose to train separately each sensor (using the basic concepts previously explained in the 'CNN for images section'). The results coming from this first step are then elaborated together in the last CNN layers, thus providing a specific classified outcome.

Figure below shows in a more understandable way the acceleration data pathway, starting from the pre-processed data (see chapter 4.5) and ending with the classification of the exercise analyzed.

More precisely, the figure shows how the whole exercise repetition (which lasts 3.5 seconds and comprises 350 samples), coming from one sensor, is used as input for a 2D CNN. The information related to each IMU is initially processed separately from the other ones. The architecture adopted consists of two convolution layers with filter numbers of 64 and 32, and sizes of (1 x 50) and (1 x 20), respectively. Each layer is followed by a MaxPooling and a Dropout layers, with drop percentage 10% and 20%, respectively. The Pooling layer reduce by a factor of 5 the convolution layer's width and height, while dropout controls the neurons overfitting. The use of pooling and dropout after each convolution layer significantly reduces the number of parameters in the fully connected layers: in this way, training becomes faster. After these first processing steps, all the results are concatenated together to pass through the last layers. The top layers in CNN are stacked by two fully connected neural networks. The latter are expected to combine different local structures in the lower layers for the final classification purpose. The SoftMax approach, eventually, allows to reach the goal of a multilabel classification. In case of Normal Squat or Wide Squat exercises, the classifier will have to choose among 6 different possibilities. As regards Deadlift, instead, the output will be only one among 4 feasible classifications. The dataset percentage to create subsets for training, testing and validation were respectively 70%, 20% and 10%.

Table 10 provides a detailed overview of the 2D CNN model.

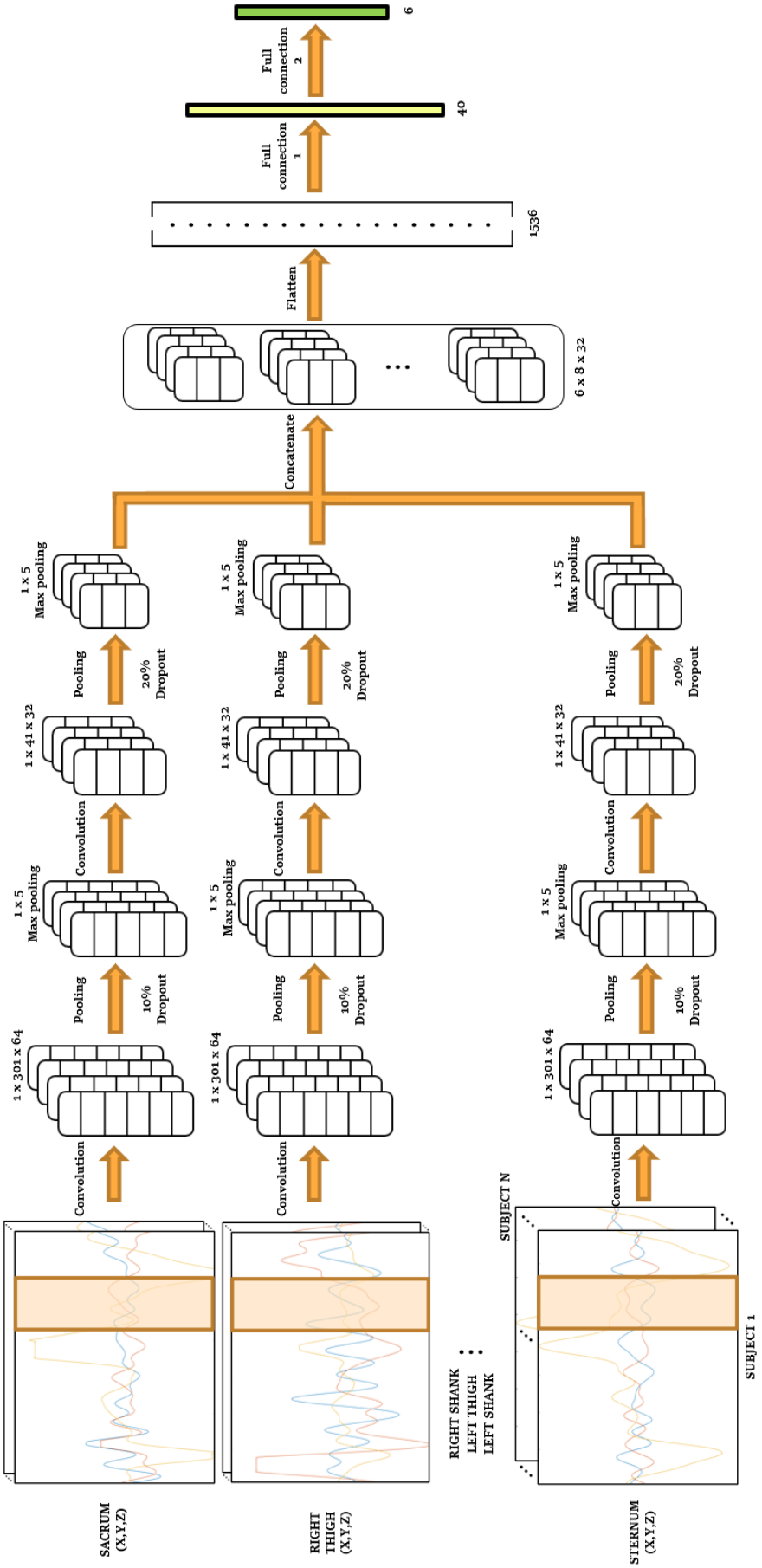

6
Full connection 2
40
Full connection 1
1536
Flatten
6 x 8 x 32
Concatenate
1 x 5 Max pooling
Pooling
20% Dropout
1 x 41 x 32
Convolution
1 x 5 Max pooling
Pooling
10% Dropout
1 x 301 x 64
Convolution
SACRUM (X,Y,Z)
RIGHT THIGH (X,Y,Z)
STERNUM (X,Y,Z)
RIGHT SHANK
LEFT THIGH
LEFT SHANK
SUBJECT N
SUBJECT 1

Table 10: the CNN was created ad-hoc to elaborate singularly the input coming from the different sensors and then combining all the results to classify the motion. The table shows the network parameters.

Model: "model_6"

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 1, 350, 3)]	0	
input_2 (InputLayer)	[(None, 1, 350, 3)]	0	
input_3 (InputLayer)	[(None, 1, 350, 3)]	0	
input_4 (InputLayer)	[(None, 1, 350, 3)]	0	
input_5 (InputLayer)	[(None, 1, 350, 3)]	0	
input_6 (InputLayer)	[(None, 1, 350, 3)]	0	
conv2d (Conv2D)	(None, 1, 301, 64)	9664	input_1[0][0]
conv2d_2 (Conv2D)	(None, 1, 301, 64)	9664	input_2[0][0]
conv2d_4 (Conv2D)	(None, 1, 301, 64)	9664	input_3[0][0]
conv2d_6 (Conv2D)	(None, 1, 301, 64)	9664	input_4[0][0]
conv2d_8 (Conv2D)	(None, 1, 301, 64)	9664	input_5[0][0]
conv2d_10 (Conv2D)	(None, 1, 301, 64)	9664	input_6[0][0]
dropout (Dropout)	(None, 1, 301, 64)	0	conv2d[0][0]
dropout_2 (Dropout)	(None, 1, 301, 64)	0	conv2d_2[0][0]
dropout_4 (Dropout)	(None, 1, 301, 64)	0	conv2d_4[0][0]
dropout_6 (Dropout)	(None, 1, 301, 64)	0	conv2d_6[0][0]
dropout_8 (Dropout)	(None, 1, 301, 64)	0	conv2d_8[0][0]
dropout_10 (Dropout)	(None, 1, 301, 64)	0	conv2d_10[0][0]
max_pooling2d (MaxPooling2D)	(None, 1, 60, 64)	0	dropout[0][0]
max_pooling2d_2 (MaxPooling2D)	(None, 1, 60, 64)	0	dropout_2[0][0]

max_pooling2d_4 (MaxPooling2D)	(None, 1, 60, 64)	0	dropout_4[0][0]
max_pooling2d_6 (MaxPooling2D)	(None, 1, 60, 64)	0	dropout_6[0][0]
max_pooling2d_8 (MaxPooling2D)	(None, 1, 60, 64)	0	dropout_8[0][0]
max_pooling2d_10 (MaxPooling2D)	(None, 1, 60, 64)	0	dropout_10[0][0]
conv2d_1 (Conv2D)	(None, 1, 41, 32)	40992	max_pooling2d[0][0]
conv2d_3 (Conv2D)	(None, 1, 41, 32)	40992	max_pooling2d_2[0][0]
conv2d_5 (Conv2D)	(None, 1, 41, 32)	40992	max_pooling2d_4[0][0]
conv2d_7 (Conv2D)	(None, 1, 41, 32)	40992	max_pooling2d_6[0][0]
conv2d_9 (Conv2D)	(None, 1, 41, 32)	40992	max_pooling2d_8[0][0]
conv2d_11 (Conv2D)	(None, 1, 41, 32)	40992	max_pooling2d_10[0][0]
dropout_1 (Dropout)	(None, 1, 41, 32)	0	conv2d_1[0][0]
dropout_3 (Dropout)	(None, 1, 41, 32)	0	conv2d_3[0][0]
dropout_5 (Dropout)	(None, 1, 41, 32)	0	conv2d_5[0][0]
dropout_7 (Dropout)	(None, 1, 41, 32)	0	conv2d_7[0][0]
dropout_9 (Dropout)	(None, 1, 41, 32)	0	conv2d_9[0][0]
dropout_11 (Dropout)	(None, 1, 41, 32)	0	conv2d_11[0][0]
max_pooling2d_1 (MaxPooling2D)	(None, 1, 8, 32)	0	dropout_1[0][0]
max_pooling2d_3 (MaxPooling2D)	(None, 1, 8, 32)	0	dropout_3[0][0]
max_pooling2d_5 (MaxPooling2D)	(None, 1, 8, 32)	0	dropout_5[0][0]
max_pooling2d_7 (MaxPooling2D)	(None, 1, 8, 32)	0	dropout_7[0][0]
max_pooling2d_9 (MaxPooling2D)	(None, 1, 8, 32)	0	dropout_9[0][0]
concatenate (Concatenate)	(None, 6, 8, 32)	0	max_pooling2d_1[0][0] max_pooling2d_3[0][0] max_pooling2d_5[0][0] max_pooling2d_7[0][0] max_pooling2d_9[0][0] max_pooling2d_11[0][0]
flatten (Flatten)	(None, 1536)	0	concatenate[0][0]
dense (Dense)	(None, 40)	61480	flatten[0][0]
dropout_12 (Dropout)	(None, 40)	0	dense[0][0]
dense_1 (Dense)	(None, 6)	246	dropout_12[0][0]

Total params: 365,662
Trainable params: 365,662
Non-trainable params: 0

CHAPTER 5
RESULTS AND DISCUSSION

In this chapter, all the outcomes obtained through the OpenSim models' comparison and the results of ML algorithms explained in Chapter 4 are reported.

Subchapter 5.1 highlights in detail the differences between the "Catelli" and "Gait2392" models through both a numerical and a graphical analysis.
Subchapter 5.2 refers to real acceleration data acquired on ground, showing results of the 6 different ML approaches exploited in this thesis.
Subchapter 5.3 focuses on the same real acceleration data collected on ground and shows the results obtained by the combination of different sensors.
Subchapter 5.4 describes the results on a customized classification for single subject, still considering the same real acceleration data saved on ground.
Subchapter 5.5 regards the comparison between real and simulated datasets.

Every result is referred to a multi-label classification basing on the three exercises analyzed: Normal Squat (NS), Wide Squat (WS) and Deadlift (ND).

5.1 OPENSIM MODEL COMPARISON

According to the tables and graphs reported below, we clearly understand and notice the differences between the Catelli and Gait2392 models. The numbers circled in red (Tables 12, 14, 16) represent ROMs that derive from maximum and/or minimum values associable with outlier (as shown in chapter 4.2, each model presents specific motion boundaries of the three joints considered, namely hip, knee and ankle). Consequently, such ROM values are not acceptable.

Table 11: each line of this numeric table shows the anonymous reference of a single subject recruited at NASA JSC. Each column refers to a specific right or left joint. The numbers refer to the ROM mean based on 4 repetitions of the same movement. In particular, here we consider the Deadlift (ND) exercise performed by the 'Catelli' model.

S/J	Hip_r	Knee_r	Ankle_r	Hip_l	Knee_l	Ankle_l
S1111	121,3697	85,69457	24,22753	113,9596	81,78905	21,60089
S2222	73,80014	67,80053	19,45827	78,45191	83,28117	27,74276
S3333	124,5267	61,22041	8,896856	123,9282	50,79806	9,943735
S4444	77,39766	87,08753	19,6552	75,67771	84,73529	18,85841
S5555	84,93948	88,21156	26,34905	95,68459	96,92443	28,32456
S6666	137,4012	87,66848	27,48915	130,4494	82,4377	18,95039

Table 12: each line of this numeric table shows the anonymous reference of a single subject recruited at NASA JSC. Each column refers to a specific right or left joint. The numbers refer to the ROM mean based on 4 repetitions of the same movement. In particular, here we consider the Deadlift (ND) exercise performed by the 'gait2392_simbody' model. The numbers circled in red represent ROMs that derive from MAXIMUM and/or MINIMUM values associable with OUTLIERS.

S/J	Hip_r	Knee_r	Ankle_r	Hip_l	Knee_l	Ankle_l
S1111	124,1894	80,99567	20,08429	130,1669	86,45322	21,28336
S2222	99,51744	74,4996	19,22199	98,71275	76,13029	21,23268
S3333	102,4101	60,68877	12,664	104,689	59,51128	13,06959
S4444	59,07564	78,59933	17,92993	60,13591	81,74657	18,6606
S5555	126,1782	105,2793	35,73585	127,651	110,4064	38,33279
S6666	102,7391	79,82655	23,27357	103,4028	79,46512	21,33342

Table 13: each line of this numeric table shows the anonymous reference of a single subject recruited at NASA JSC. Each column refers to a specific right or left joint. The numbers refer to the ROM mean based on 4 repetitions of the same movement. In particular, here we consider the Wide Squat (WS) exercise performed by the 'Catelli' model.

S/J	Hip_r	Knee_r	Ankle_r	Hip_l	Knee_l	Ankle_l
S1111	123,7167	124,52	26,41688	105,3294	123,3755	39,16017
S2222	99,87494	124,1925	34,2398	102,5267	128,2763	35,64979
S3333	121,7394	96,14023	23,8693	131,5895	86,08414	13,80878
S4444	67,40584	100,9814	33,05694	68,67526	98,39535	30,37395
S5555	118,2712	96,70113	25,70824	127,2433	103,776	31,84618
S6666	128,2184	107,8393	55,02301	135,9329	110,33	52,65335

Table 14: each line of this numeric table shows the anonymous reference of a single subject recruited at NASA JSC. Each column refers to a specific right or left joint. The numbers refer to the ROM mean based on 4 repetitions of the same movement. In particular, here we consider the Wide Squat (WS) exercise performed by the 'gait2392_simbody' model. The numbers circled in red represent ROMs that derive from MAXIMUM and/or MINIMUM values associable with OUTLIERS.

S/J	Hip_r	Knee_r	Ankle_r	Hip_l	Knee_l	Ankle_l
S1111	114,8234	114,2059	31,20674	125,7856	120,1248	41,29895
S2222	101,8718	104,2775	37,37798	94,94686	101,7499	37,45867
S3333	72,59425	75,78262	17,52938	85,51717	87,02095	19,44688
S4444	75,04299	87,56794	36,04904	86,60868	87,99195	39,75728
S5555	187,2488	78,03863	43,00216	244,7384	102,3213	94,36935
S6666	61,37522	111,4349	77,93621	73,49963	95,75962	63,88159

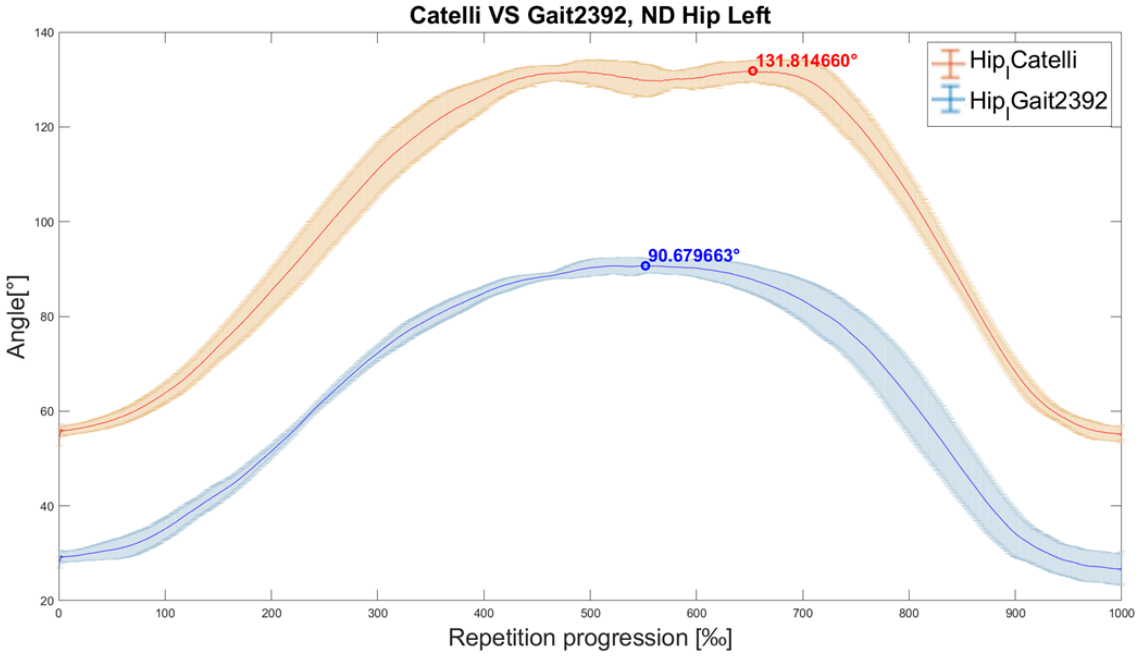
Table 15: each line of this numeric table shows the anonymous reference of a single subject recruited at NASA JSC. Each column refers to a specific right or left joint. The numbers refer to the ROM mean based on 4 repetitions of the same movement. In particular, here we consider the Normal Squat (NS) exercise performed by the 'Catelli' model. The value circled in red is due to a lack of data associated with the ankle.

S/J	Hip_r	Knee_r	Ankle_r	Hip_l	Knee_l	Ankle_l
S1111	119,2426	120,2784	40,47931	104,4016	117,3379	41,7785
S2222	83,73391	116,4687	34,84377	85,77277	119,443	39,16649
S3333	121,1151	90,4783	0	124,5393	76,54788	17,12472
S4444	64,85278	100,5287	32,90775	68,11579	96,94822	28,49765
S5555	129,3637	103,5724	27,71057	119,6242	103,5604	31,5792
S6666	111,8943	116,6821	37,16484	105,0169	113,5155	38,42664

Table 16: each line of this numeric table shows the anonymous reference of a single subject recruited at NASA JSC. Each column refers to a specific right or left joint. The numbers refer to the ROM mean based on 4 repetitions of the same movement. In particular, here we consider the Normal Squat (NS) exercise performed by the 'gait2392_simbody' model. The numbers circled in red represent ROMs that derive from MAXIMUM and/or MINIMUM values associable with OUTLIERS.

S/J	Hip_r	Knee_r	Ankle_r	Hip_l	Knee_l	Ankle_l
S1111	121,7551	122,3504	45,1028	130,3464	126,1203	45,909
S2222	98,47211	121,392	41,58637	92,76622	117,4148	41,42287
S3333	117,9074	86,35304	27,03508	115,9997	84,48931	29,55623
S4444	76,54	79,65568	32,96138	95,37018	77,76333	34,2759
S5555	128,6341	110,1619	35,16935	125,5943	113,1243	40,64575
S6666	134,4115	127,1456	43,21188	118,5623	122,5975	48,12429

To have a clearer view about the models' differences, the following graphs show the variation of both the left and right joints angle with respect to the repetition progression of the Normal Deadlift (ND) and Normal Squat (NS) exercises. In particular, the blue and orange lines and ranges represent respectively the mean (based on 4 repetitions of a single subject) and the standard deviation.



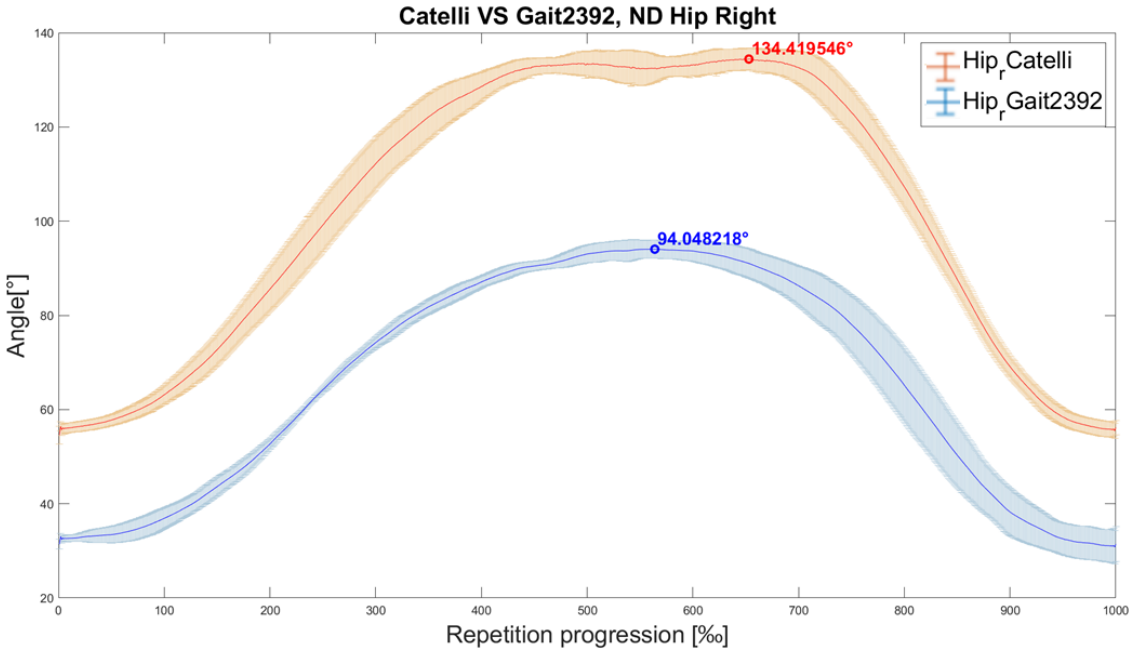
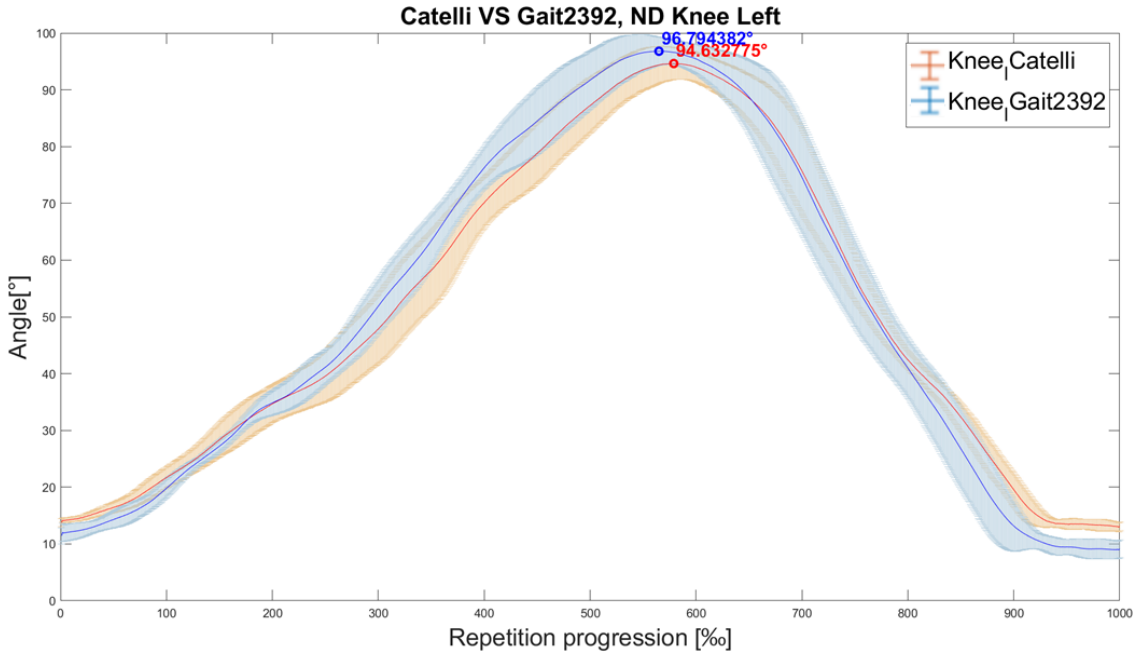


Figure 68: the two graphs above show respectively the variation of both the left and right Hip joint angle with respect to the repetition progression of the Normal Deadlift (ND) exercise. In particular, the blue and orange lines represent the mean based on 4 repetitions of the same motion, while the orange and blue ranges correspond to the standard deviation. The figure also shows the Maximum value of the two curves. This approach highlights the differences between the 'Catelli' and 'gait2392_simbody'.



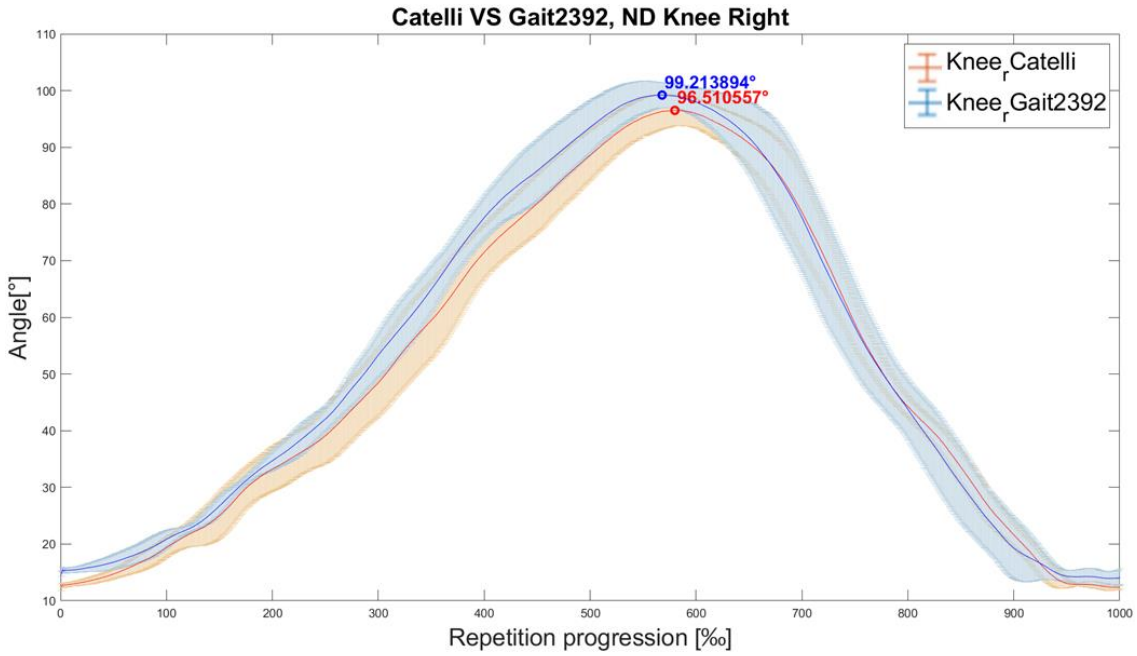


Figure 69: the two graphs above show respectively the variation of both the left and right Knee joint angle with respect to the repetition progression of the Normal Deadlift (ND) exercise. In particular, the blue and orange lines represent the mean based on 4 repetitions of the same motion, while the orange and blue ranges correspond to the standard deviation. The figure also shows the Maximum value of the two curves. This approach highlights the differences between the 'Catelli' and 'gait2392_simbody' models.

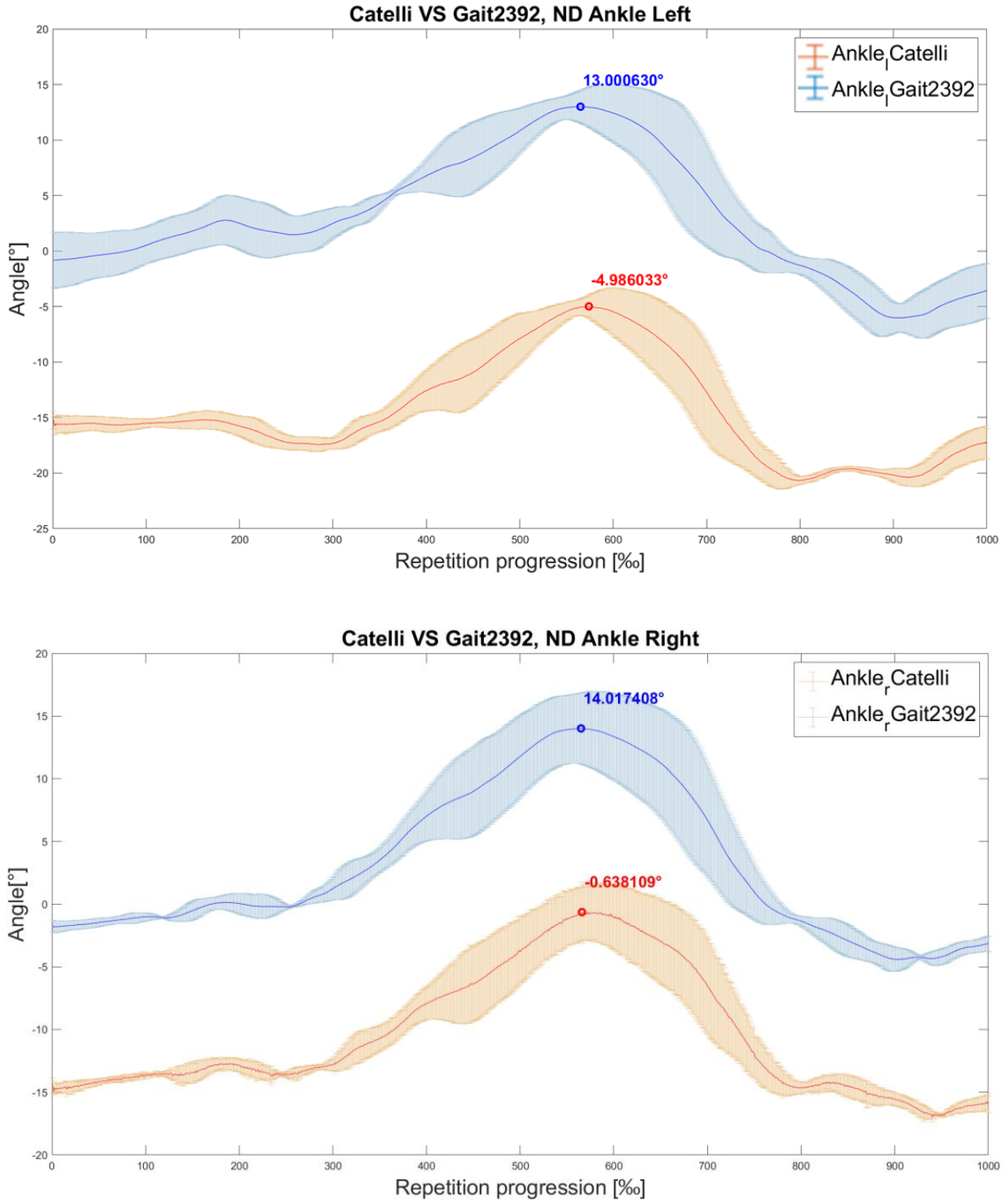


Figure 70: the two graphs above show respectively the variation of both the left and right Ankle joint angle with respect to the repetition progression of the Normal Deadlift (ND) exercise. In particular, the blue and orange lines represent the mean based on 4 repetitions of the same motion, while the orange and blue ranges correspond to the standard deviation. The figure also shows the Maximum value of the two curves. This approach highlights the differences between the 'Catelli' and 'gait2392_simbody' models.

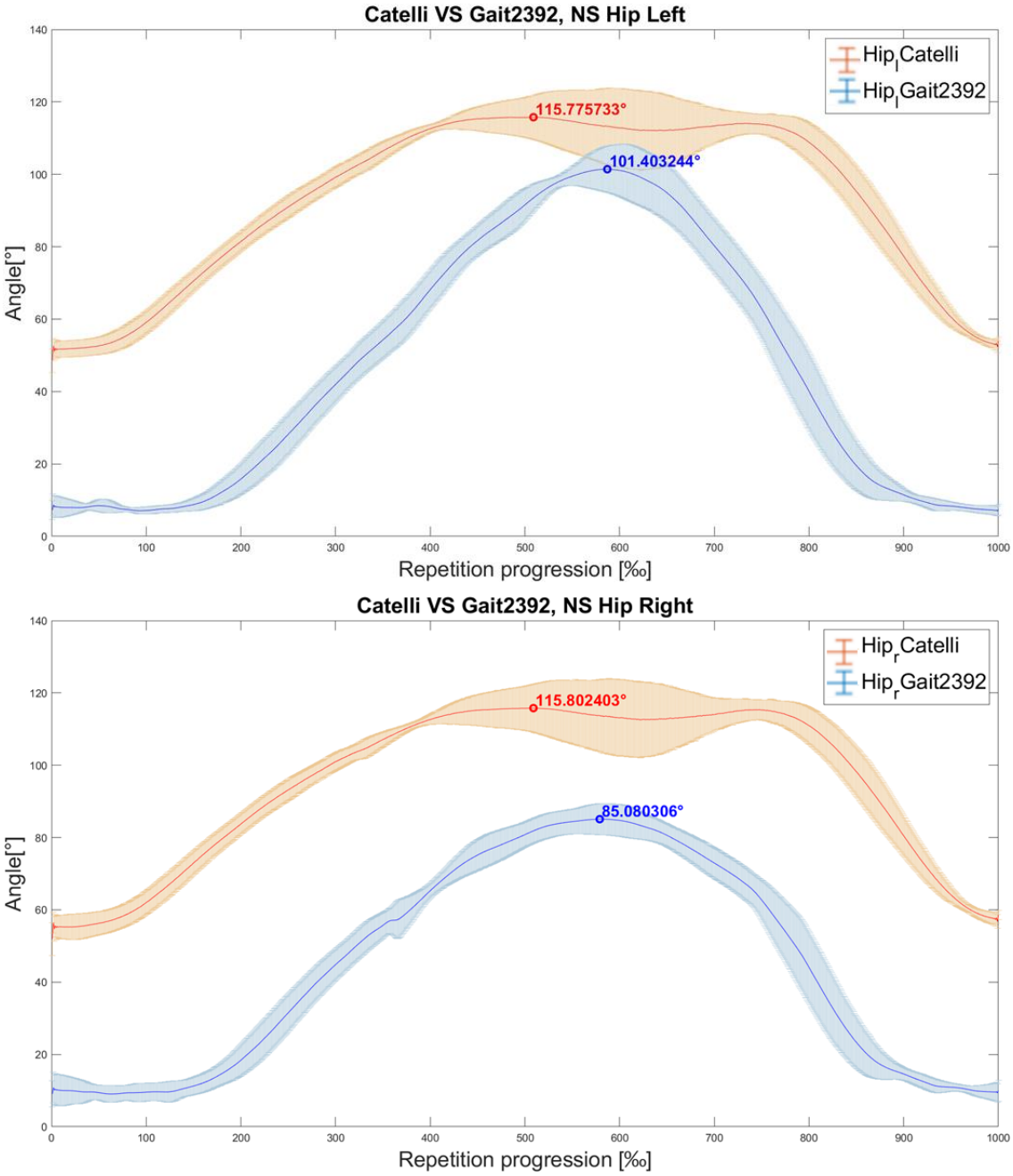
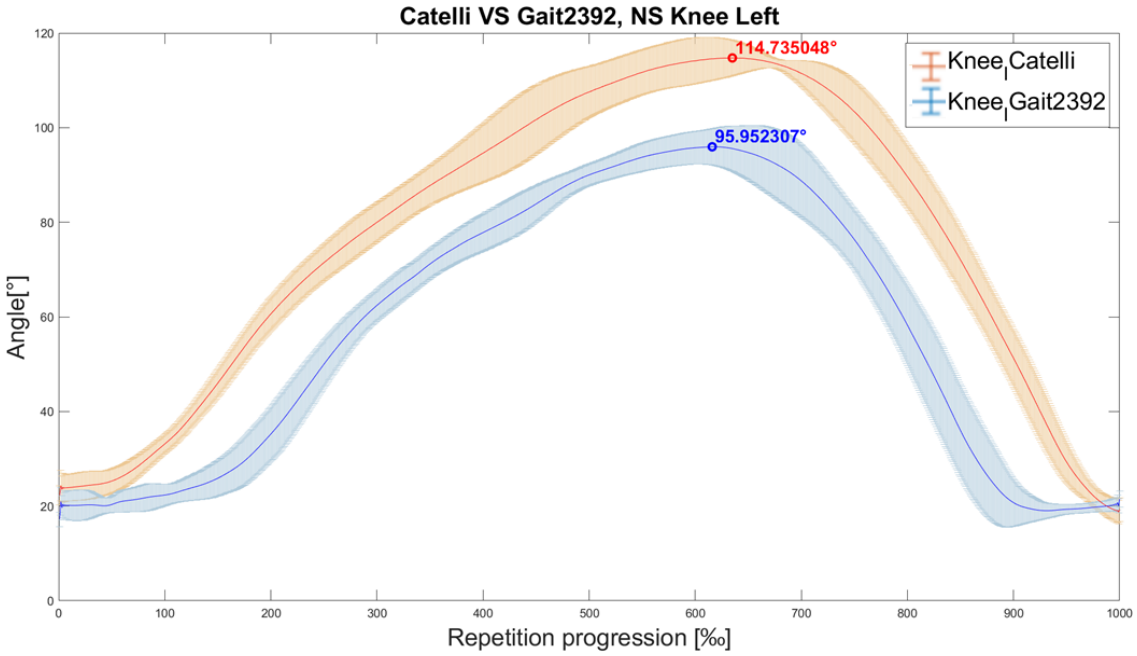


Figure 71: the two graphs above show respectively the variation of both the left and right Hip joint angle with respect to the repetition progression of the Normal Squat (NS) exercise. In particular, the blue and orange lines represent the mean based on 4 repetitions of the same motion, while the orange and blue ranges correspond to the standard deviation. The figure also shows the Maximum value of the two curves. This approach highlights the differences between the 'Catelli' and 'gait2392_simbody' models.



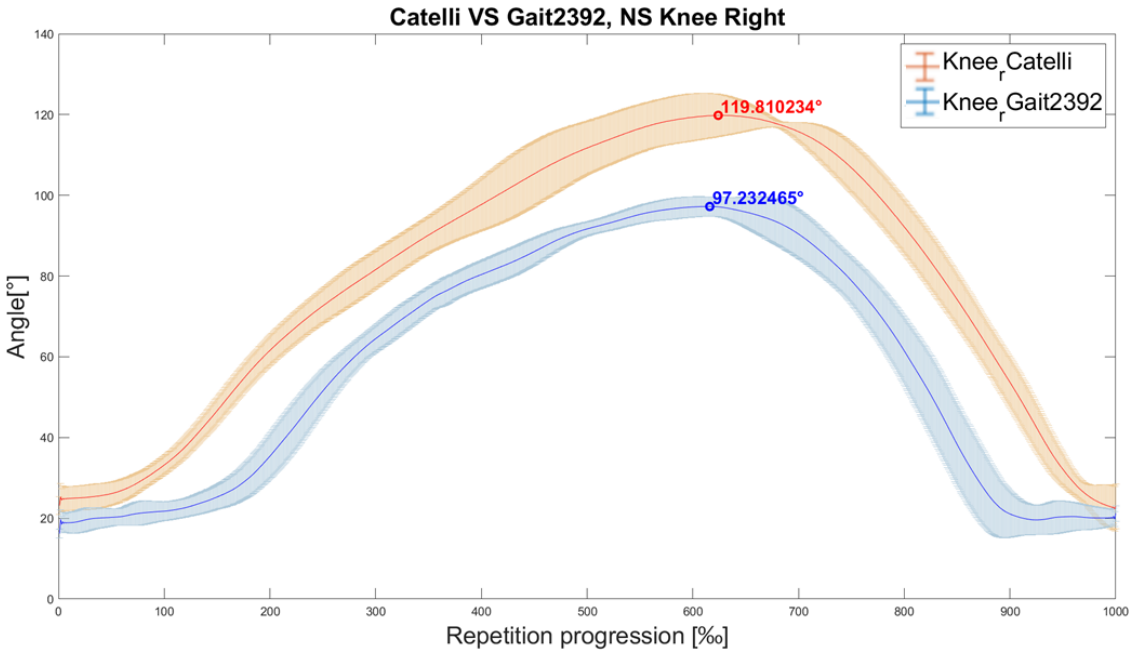


Figure 72: the two graphs above show respectively the variation of both the left and right Knee joint angle with respect to the repetition progression of the Normal Squat (NS) exercise. In particular, the blue and orange lines represent the mean based on 4 repetitions of the same motion, while the orange and blue ranges correspond to the standard deviation. The figure also shows the Maximum value of the two curves. This approach highlights the differences between the 'Catelli' and 'gait2392_simbody' models.

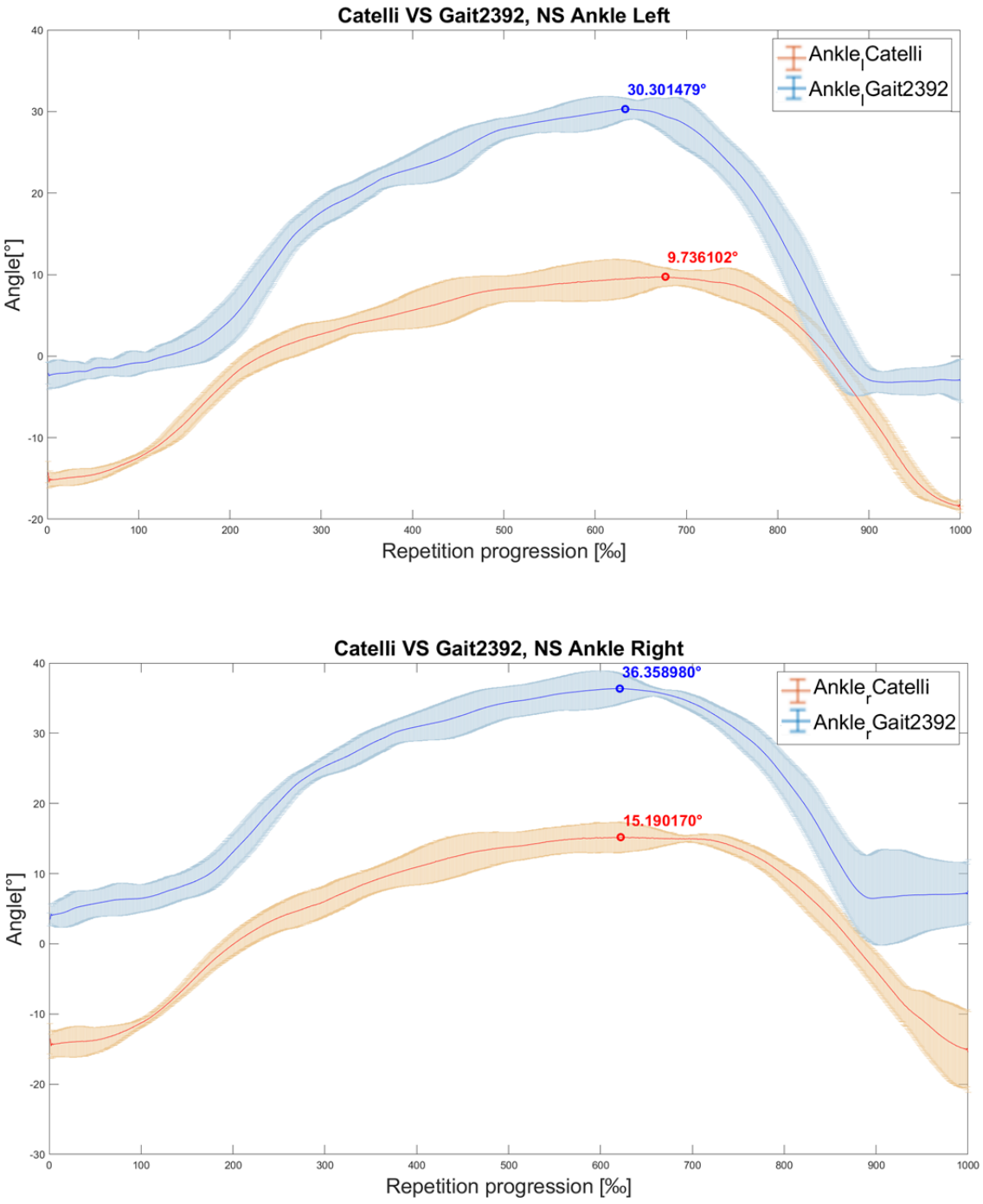


Figure 73: the two graphs above show respectively the variation of both the left and right Ankle joint angle with respect to the repetition progression of the Normal Squat (NS) exercise. In particular, the blue and orange lines represent the mean based on 4 repetitions of the same motion, while the orange and blue ranges correspond to the standard deviation. The figure also shows the Maximum value of the two curves. This approach highlights the differences between the 'Catelli' and 'gait2392_simbody' models.

5.2 IMU DATA ANALYSIS AND RESULTS

A summary of the accuracies coming from the six algorithms chosen can be found in Table 17. All the results are referred to the cooperation of the six IMU sensors (shown in chapter 4).

Table 17: summary containing all the accuracies obtained by the six different Machine Learning approaches. In general, SVM and CNN reached better results with respect to the other methods chosen.

	CNN	SVM	KNN	DT	XGB	MLP
NS	84.30%	87.79%	77.91%	56.98%	80.23%	72.09%
WS	86.91%	86.38%	67.54%	47.12%	80.10%	70.68%
DL	82.05%	83.76%	73.50%	42.74%	75.21%	68.38%

5.2.1 NORMAL SQUAT

The original dataset, containing 1314 features for six sensors (hence, 219 features for each of them), is reduced with PCA to 198 features (the ones which represent at least 90% of variance). The algorithms have been trained with 70% of data, tested with 20% and validated with the remaining 10%. A single entry corresponded to one repetition that was opportunely labeled as a number between zero and five {0 1 2 3 4 5}, where 0 corresponded to a correct exercise (CO), 1 to Knees Over Toes (KOT), 2 to Valgus Knees (Kv), 3 to Rounded Back (Rb), 4 to Raised Heels (Rh), 5 to Shallow (Sh).

A total of 792 observations were used, where CO repetitions were 143, KOT repetitions were 113, Kv repetitions were 127, Rb repetitions were 139, Rh repetitions were 129 and Sh repetitions were 141.

Statistical parameters (precision, recall, f1-score), confusion matrixes and graphs resulting from each testing phase are shown below. It was observed greater accuracies for the SVM (87.79 %) and CNN (84.30%) algorithms with respect to the

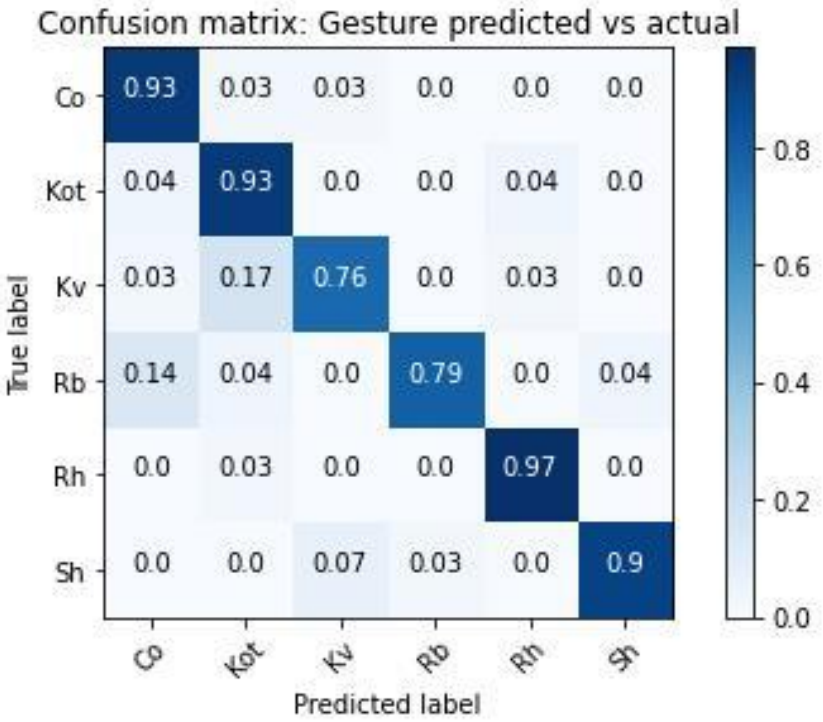
other ones. Below only SVM and CNN are reported since the other approaches present less accurate outcomes (see Appendix B).

Support vector machine (SVM) results

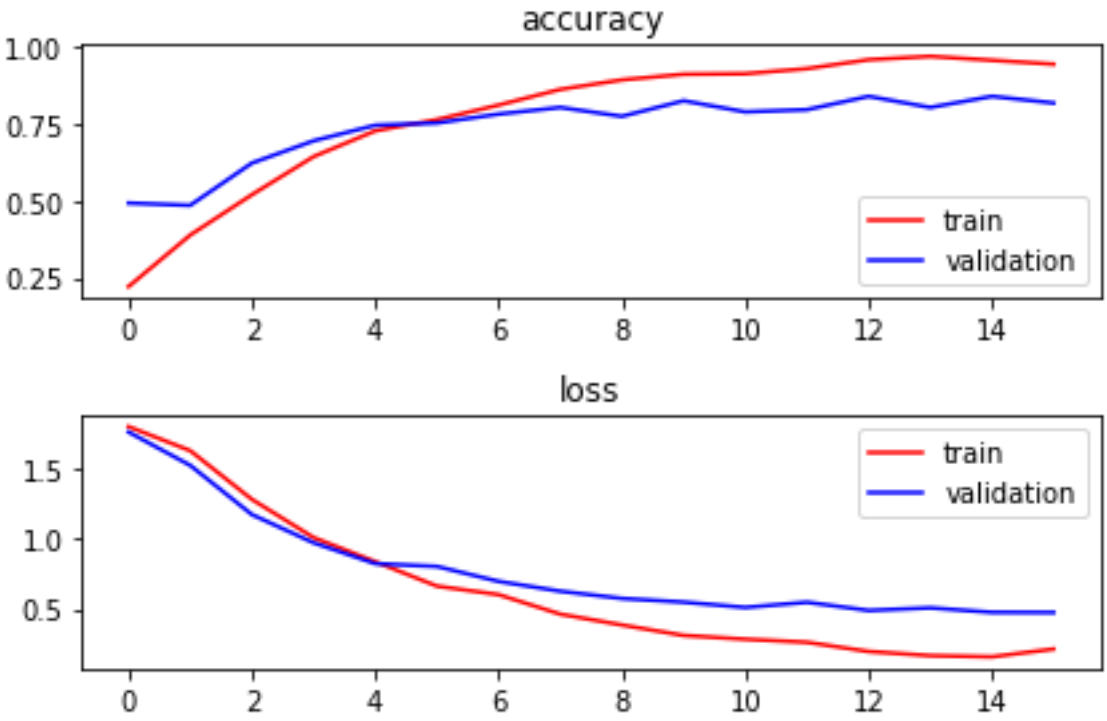
Test set report

	precision	recall	f1-score	support
Co	0.82	0.93	0.87	29
Kot	0.76	0.93	0.84	28
Kv	0.88	0.76	0.81	29
Rb	0.96	0.79	0.86	28
Rh	0.93	0.97	0.95	29
Sh	0.96	0.90	0.93	29
accuracy			0.88	172
macro avg	0.89	0.88	0.88	172
weighted avg	0.89	0.88	0.88	172

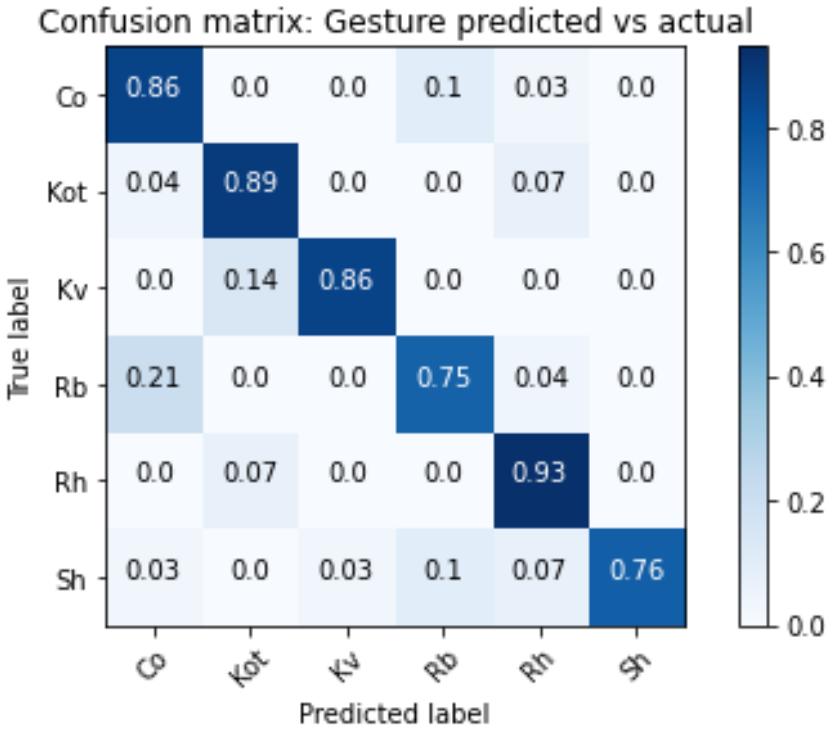
Accuracy: 87.79%



Convolutional Neural Network (CNN) results



Accuracy: 84.30 %



5.2.2 WIDE SQUAT

The original dataset, containing 1314 features for six sensors (hence, 219 features for each of them), is reduced with PCA to 216 features (the ones which represent at least 90% of variance). The algorithms have been trained with 70% of data, tested with 20% and validated with the remaining 10%. A single entry corresponded to one repetition that was opportunely labeled as a number between zero and five {0 1 2 3 4 5}, where 0 corresponded to a correct exercise (CO), 1 to Knees Over Toes (KOT), 2 to Valgus Knees (Kv), 3 to Rounded Back (Rb), 4 to Raised Heels (Rh), 5 to Shallow (Sh).

A total of 793 observations were used, where CO repetitions were 159, KOT repetitions were 119, Kv repetitions were 123, Rb repetitions were 133, Rh repetitions were 123 and Sh repetitions were 136.

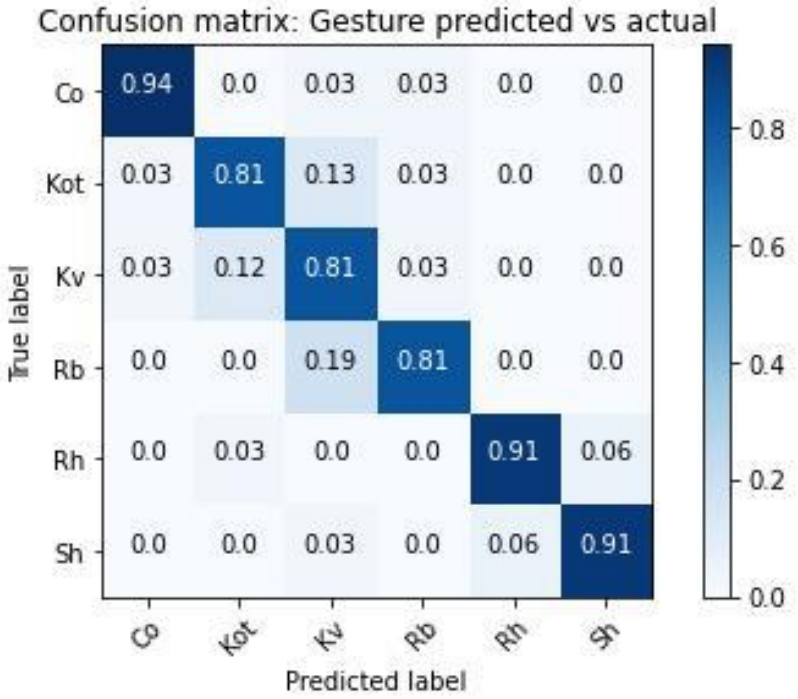
Statistical parameters (precision, recall, f1-score), confusion matrixes and graphs resulting from each testing phase are shown below. It was observed greater accuracies for the SVM (86.38 %) and CNN (86.91%) algorithms with respect to the other ones. Below only SVM and CNN are reported since the other approaches present less accurate outcomes (see Appendix B).

Support vector machine (SVM) results

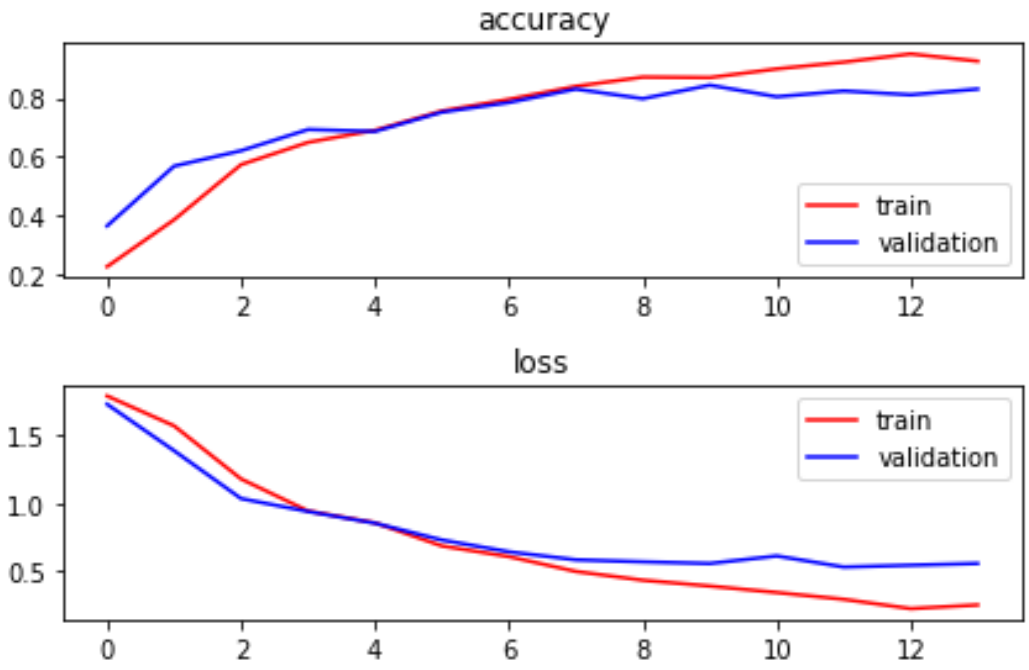
Test set report

	precision	recall	f1-score	support
CO	0.94	0.94	0.94	32
Kot	0.79	0.87	0.83	31
Kv	0.80	0.75	0.77	32
Rb	0.83	0.94	0.88	32
Rh	0.93	0.84	0.89	32
Sh	0.90	0.84	0.87	32
accuracy			0.86	191
macro avg	0.87	0.86	0.86	191
weighted avg	0.87	0.86	0.86	191

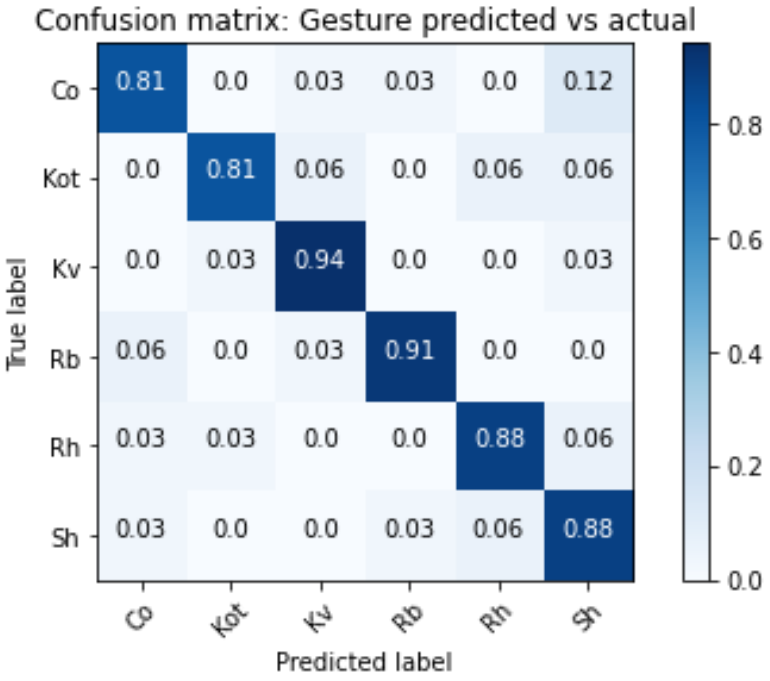
Accuracy: 86.38%



Convolutional Neural Network (CNN) results



Accuracy: 86.91%



5.2.3 DEADLIFT

The original dataset, containing 1314 features for six sensors (hence, 219 features for each of them), is reduced with PCA to 152 features (the ones which represent at least 90% of variance). The algorithms have been trained with 70% of data, tested with 20% and validated with the remaining 10%. A single entry corresponded to one repetition that was opportunely labeled as a number between zero and three {0 1 2 3}, where 0 corresponded to a Correct Exercise (CO), 1 to Bar Over Shoulders (BOS), 2 to Rounded back (Rb), 3 to Hyperextended back (Hb).

A total of 508 observations were used, where CO repetitions were 146, BOS repetitions were 103, Rb repetitions were 132 and Hb repetitions were 127.

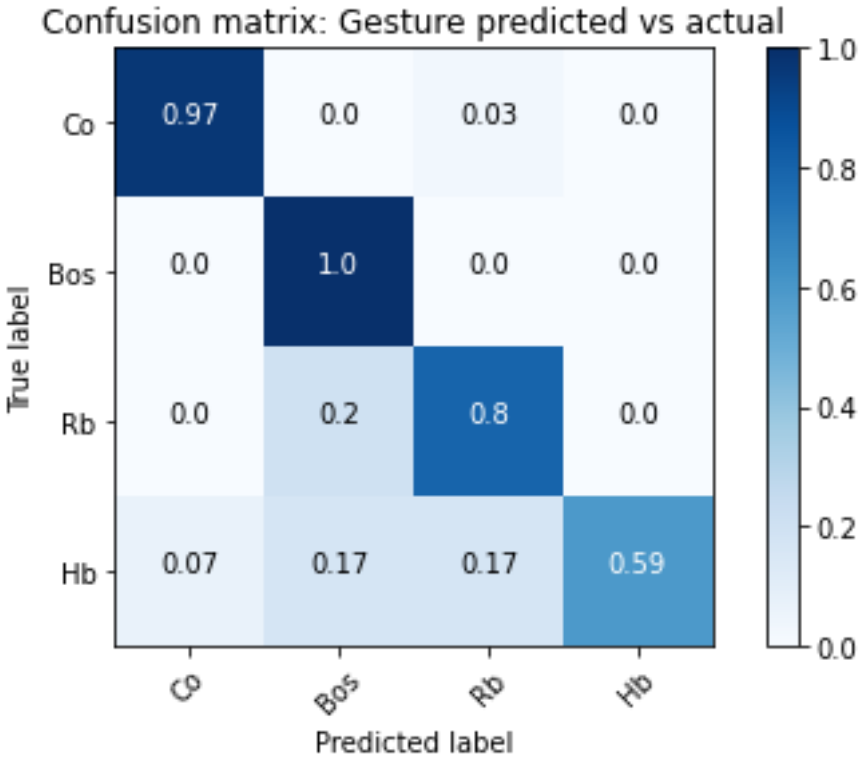
Statistical parameters (precision, recall, f1-score), confusion matrixes and graphs resulting from each testing phase are shown below. It was observed greater accuracies for the SVM (83.76 %) and CNN (82.05%) algorithms with respect to the other ones. Below only SVM and CNN are reported since the other approaches present less accurate outcomes (see Appendix B).

Support vector machine (SVM) results

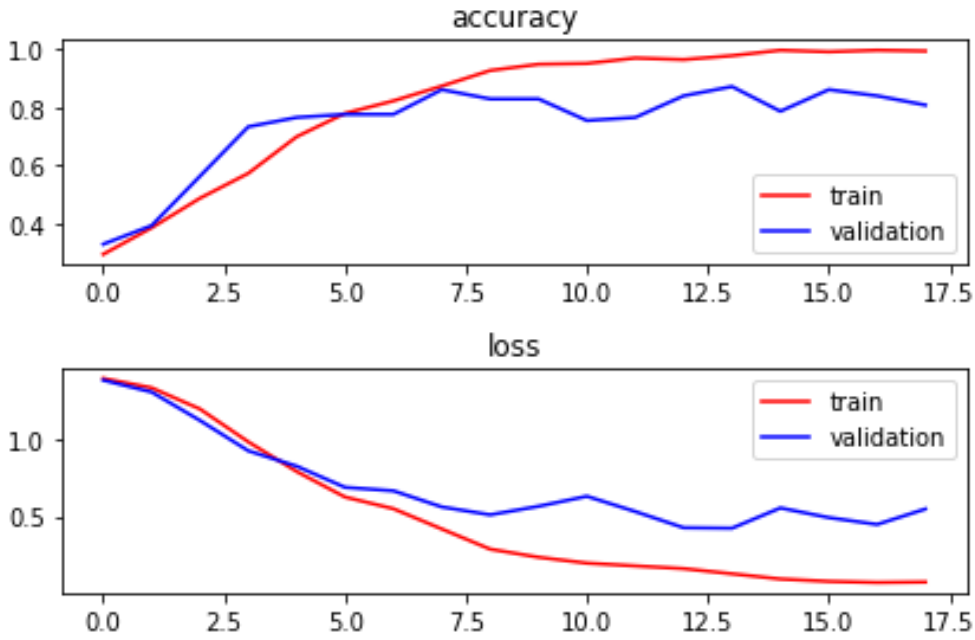
Test set report

	precision	recall	f1-score	support
Co	0.93	0.97	0.95	29
Bos	0.72	1.00	0.84	29
Rb	0.80	0.80	0.80	30
Hb	1.00	0.59	0.74	29
accuracy			0.84	117
macro avg	0.86	0.84	0.83	117
weighted avg	0.86	0.84	0.83	117

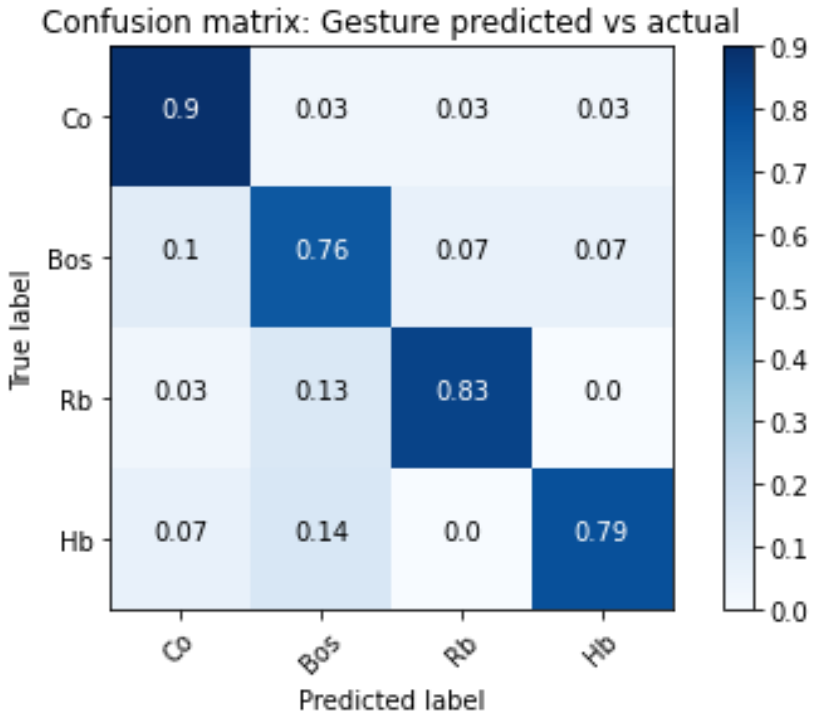
Accuracy: 83.76%



Convolutional Neural Network (CNN) results



Accuracy: 82.05%



5.3 DIFFERENT SENSORS COMBINATIONS

The current works strongly recommend the use of CNN network (see Conclusions explained in chapter 6 for further information). Since the MARS-PRE projects aims at investigating which is the minimum number of sensors (together with their specific body placement) to be exploited by the astronauts on ISS, we tested different combination of sensors. The results, in terms of accuracies, while considering the Normal Squat, Wide Squat and Deadlift exercises are reported, respectively, in tables 18, 19, 20. Better outcomes are reached when considering all the six sensors (Sacrum, Right Shank, Left Shank, Right Thigh, Left Thigh, Sternum): 84.30% (NS), 86.91% (WS) and 82.05% (DL) accuracy values. However, good results (83.25%, 81.68% and 82.05%) were obtained also with the cooperation of homolateral sensors (such as Sacrum, Left Shank, Left Thigh and Sternum) and sensors (Right Shank, Left Shank, Right Thigh, Left Thigh) placed on the lower limbs (84.29%, 81.68% and 76.92%).

Table 18: CNN accuracy results obtained by different combination of sensors when performing Normal Squat exercise.

Normal Squat - Markers' Combination	Accuracy
Sacrum, Right Shank, Left Shank	74.87%
Right Thigh, Left Thigh	80.10%
Sacrum, Left Shank, Left Thigh, Sternum	83.25%
Sacrum, Right Shank, Right Thigh, Sternum	80.63%
Sacrum, Right Thigh, Left Thigh	82.22%
Right Shank, Left Shank, Sternum	74.87%
Right Thigh, Left Thigh, Sternum	81.68%
Sacrum, Sternum	69.63%
Right Shank, Left Shank, Right Thigh, Left Thigh	84.29%
Sacrum, Right Shank, Left Shank, Right Thigh, Left Thigh, Sternum	84.30%

Table 19: CNN accuracy results obtained by different combination of sensors when performing Wide Squat exercise.

Wide Squat - Markers' Combination	Accuracy
Sacrum, Right Shank, Left Shank	78.53%
Right Thigh, Left Thigh	79.58%
Sacrum, Left Shank, Left Thigh, Sternum	81.68%
Sacrum, Right Shank, Right Thigh, Sternum	70.68%
Sacrum, Right Thigh, Left Thigh	80.63%
Right Shank, Left Shank, Sternum	69.63%
Right Thigh, Left Thigh, Sternum	80.63%
Sacrum, Sternum	65.97%
Right Shank, Left Shank, Right Thigh, Left Thigh	81.68%
Sacrum, Right Shank, Left Shank, Right Thigh, Left Thigh, Sternum	86.91%

Table 20: CNN accuracy results obtained by different combination of sensors when performing Deadlift exercise.

Deadlift - Markers' Combination	Accuracy
Sacrum, Right Shank, Left Shank	72.65%
Right Thigh, Left Thigh	73.50%
Sacrum, Left Shank, Left Thigh, Sternum	82.05%
Sacrum, Right Shank, Right Thigh, Sternum	81.20%
Sacrum, Right Thigh, Left Thigh	82.05%
Right Shank, Left Shank, Sternum	72.65%
Right Thigh, Left Thigh, Sternum	75.21%
Sacrum, Sternum	70.09%
Right Shank, Left Shank, Right Thigh, Left Thigh	76.92%
Sacrum, Right Shank, Left Shank, Right Thigh, Left Thigh, Sternum	82.05%

5.4 CUSTOMIZED CLASSIFIER FOR SINGLE SUBJECT – Examples

Still considering the IMU dataset (explained in chapter 4), the current thesis also employed the ML algorithms to build a customized classifier suited for a single person (since customization is one of the MARS-PRE project's aims). More specifically, we tried to isolate the subjects (among the 17 available) who performed the highest number of repetitions and we studied them separately. Table 21 shows the number of observations of 3 subjects performing Normal Squat (NS), Wide Squat (WS) and Deadlift (DL). Even in this case, we focused on and trained only CNNs and SVMs algorithms. Table 22 shows a summary of the results, in terms of accuracy, obtained for each single subject. See Appendix B for the related confusion matrices.

Table 21: number of single subjects' observations.

	NS						
	N. Observations	**CO**	**KOT**	**Kv**	**Rb**	**Rh**	**Sh**
S06	75	15	-	15	15	15	15
S08	104	20	15	14	20	15	20
S11	58	10	8	10	10	10	10

	WS						
	N. Observations	**CO**	**KOT**	**Kv**	**Rb**	**Rh**	**Sh**
S06	59	10	10	10	9	10	10
S08	105	20	15	15	20	15	20
S11	56	10	8	10	10	10	8

	DL				
	N. Observations	**CO**	**Bos**	**Rb**	**Hb**
S06	60	20	10	15	15
S08	68	19	11	20	18
S11	36	9	10	9	8

Table 22: accuracy results summary for customized classifiers on single subjects.

	S06		S08		S11	
	CNN	SVM	CNN	SVM	CNN	SVM
NS	100.00%	100.00%	95.83%	95.83%	91.67%	100.00%
WS	91.67%	100.00%	87.50%	95.83%	83.33%	100.00%
DL	93.75%	93.75%	93.75%	93.75%	75.00%	87.50%

5.5 SIMULATED DATA ANALYSIS AND RESULTS

While all the previous analyses were related only to IMU sensors, we tried to understand whether it was possible to classify the simulated data (i.e. those coming from the OpenSim software) starting from a classifier trained with real observations. However, comparing the latter without gravity (figure 74) and the simulated ones (figure 75), an intuitive and easily visible incompatibility is noted. Indeed, in the simulated data it is possible to observe the single repetitions simply by looking at the acceleration trend relative to the Y axis (differently from the real ones). Moreover, no machine learning algorithm was able to correctly classify the movement since the trials performed with every classifier led to meaningless results in terms of prediction accuracies (see appendix B).

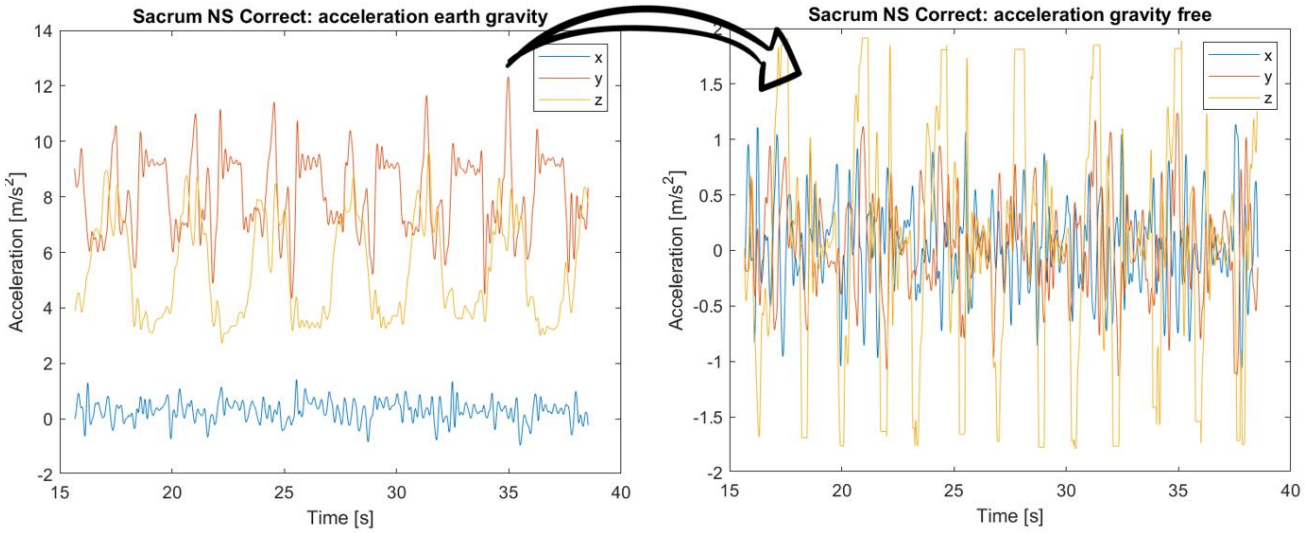


Figure 74: IMU data with gravity of a Normal Squat exercise based on a sensor placed on the Sacrum (left). Corresponding IMU data without gravity (right).

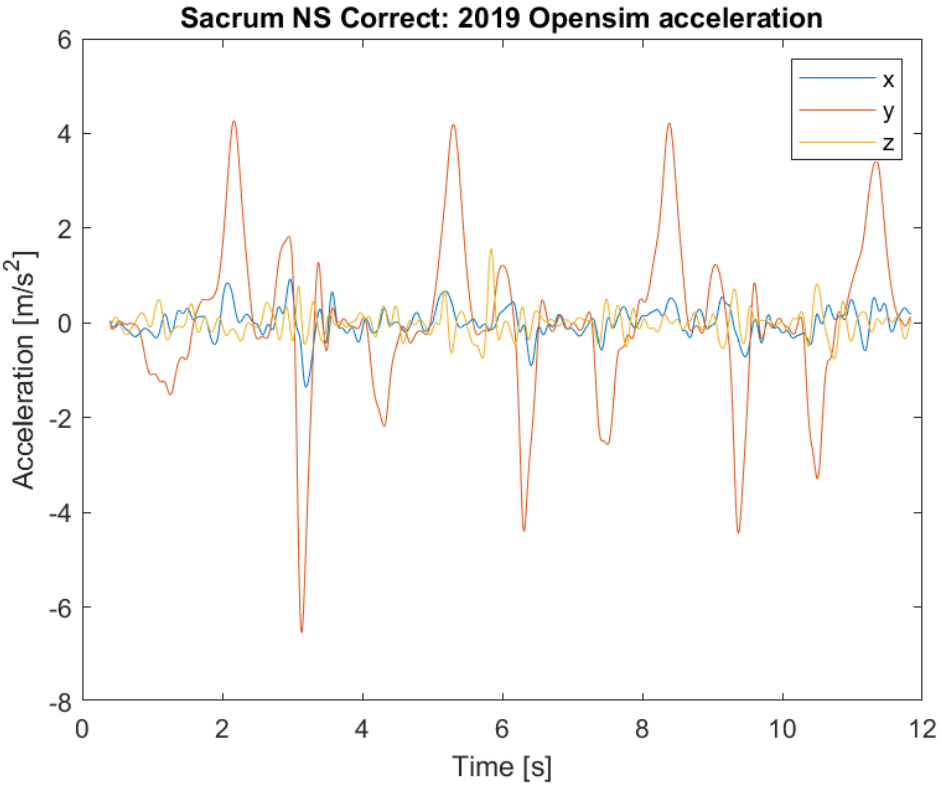


Figure 75: example of OpenSim simulated data without gravity related to a Normal Squat exercise based on a biomarker positioned on the Sacrum.

As a consequence, we have decided to treat the simulated data from OpenSim and the real IMU ones SEPARATELY. Table 23 shows the few number of observations collected in Politecnico di Milano of 2 subjects (1 male and 1 female) performing Normal Squat, Wide Squat and Deadlift. Even in this case, we focused on and trained only CNNs and SVMs algorithms.

Table 23: number of simulated data observations.

NS						
N. Observations	**CO**	**KOT**	**Kv**	**Rb**	**Rh**	**Sh**
38	8	6	6	6	6	6

WS						
N. Observations	**CO**	**KOT**	**Kv**	**Rb**	**Rh**	**Sh**
27	8	7	6	6	-	-

DL				
N. Observations	**CO**	**Bos**	**Rb**	**Hb**
25	7	6	6	6

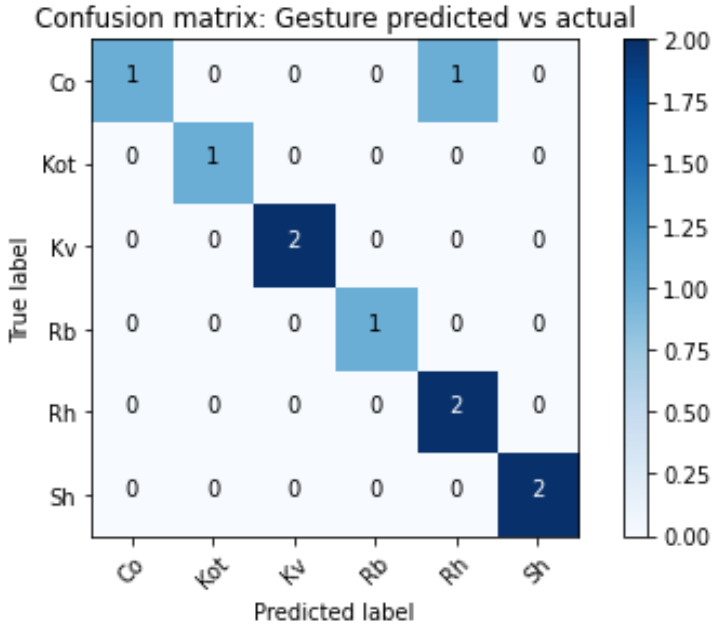
A summary of the accuracies coming from SVM and CNN algorithms can be found in Table 24.

Table 24: summary containing all the accuracies obtained by CNN and SVM algorithms when training and testing only with simulated OpenSim data.

	CNN	SVM
NS	80.00%	90.00%
WS	71.43%	85.71%
DL	75.00%	75.00%

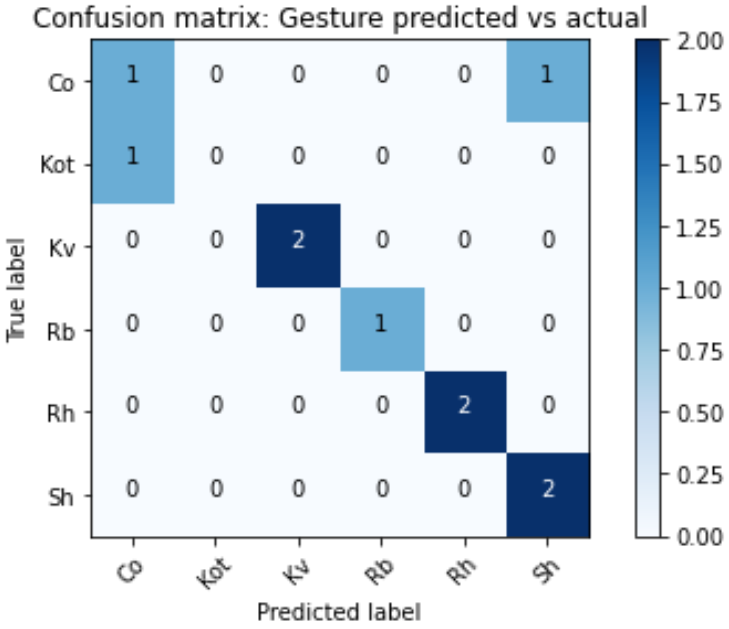
5.5.1 NORMAL SQUAT

Support vector machine (SVM) results



Accuracy: 90.00%

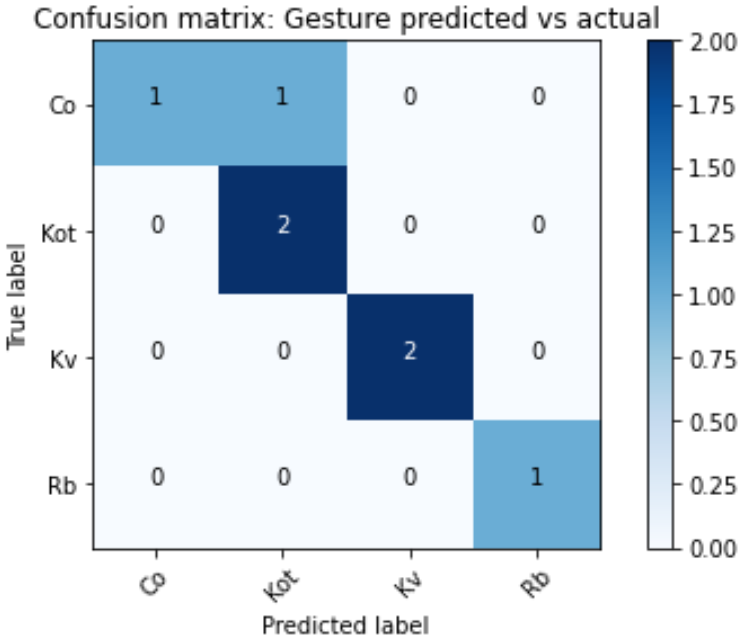
Convolutional Neural Network (CNN) results



Accuracy: 80.00%

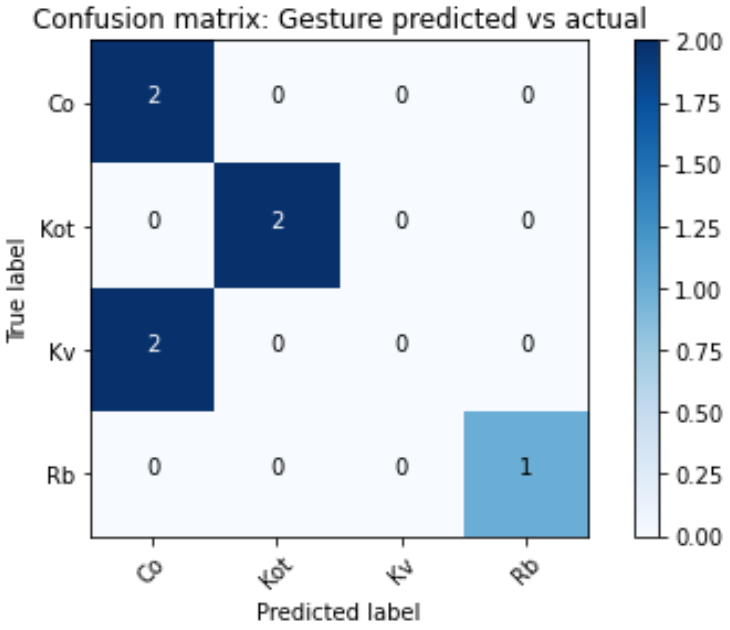
5.5.2 WIDE SQUAT

Support vector machine (SVM) results



Accuracy: 85.71%

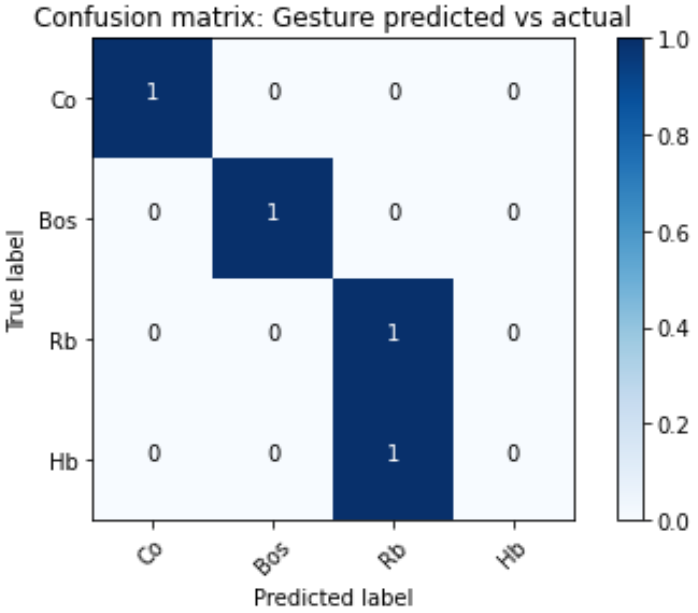
Convolutional Neural Network (CNN) results



Accuracy: 71.43%

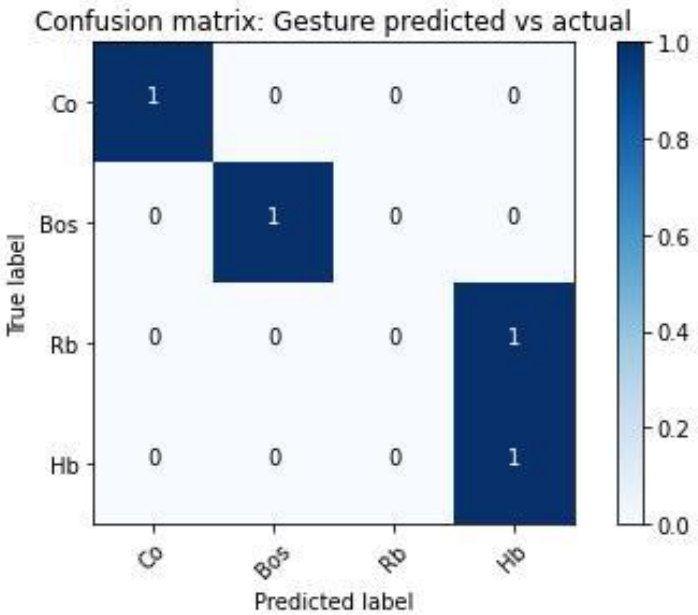
5.5.3 DEADLIFT

Support vector machine (SVM) results



Accuracy: 75.00%

Convolutional Neural Network (CNN) results



Accuracy: 75.00%

CHAPTER 6
CONCLUSIONS & FUTURE WORKS

In this work, an OpenSim biomechanical model was used to simulate countermeasure target exercises performed on ISS by astronauts during their missions. Training is essential to reduce the effects of weightlessness on musculoskeletal system deconditioning. However, dangerous or hazardous consequences can occur even when executing incorrect motion.

Keeping in mind the results obtained in Chapter 5, the conclusions of this whole project can be summarized as follows:

- The "Catelli" OpenSim biomechanical model, differently from "gait2392", demonstrated to be a valid candidate in the ARED-K research project since it allows to reach, especially regarding the knee joint, those Range of Motions (ROM) suitable for the specific training exercises (Normal Squat, Wide Squat, Deadlift) performed on ISS.

- Six different Machine Learning approaches have been considered, but only SVM and CNN allowed to reach discrete and reliable results in terms of accuracy. However, the following statements are going to make the reader understand the reason why we mainly focused on the CNN algorithm:

A) In literature, the last 6-year papers (e.g., Gemert et al. (2019), Jaehyun Lee et al. (2020); O'Reilly et al. (2015), O'Reilly et al. (2017); Avci et al. (2017), Shrestha et al. (2020), Kiranyaz et al. (2019)) strongly recommend the use of CNN to classify human motion, even for real time applications. Also, the computational speed has been proved to be higher with respect to the SVM.
B) While the SVM algorithm needs to execute longer steps in pre-processing (e.g. features extraction and feature selection with PCA), the CNN takes as input just the raw data (see chapter 4 for further information), thus lowering the total computational cost.

- Relying on the results obtained through CNN (84.30% accuracy for Normal Squat classification; 86.91% accuracy for Wide Squat classification; 82.05% accuracy for Deadlift classification), it can be inferred that our biomarkers are Sacrum, Left Thigh, Right Thigh, Left Shank, Right Shank and Sternum.

- Using CNN, we analyzed different outputs coming from several combination of sensors (see Chapter 5.3). The best results were obtained when considering all the six sensors used in the data acquisition step. However, discrete outcomes (83.25% for NS, 81.68% for WS and 82.05% for DL) were reached even with sensors placed in homolateral position (for instance, the left side of human body related to the Sacrum, Left Thigh, Left Shank and Sternum body points). Moreover, the outcomes also show that considering the sensors placed only on the lower limbs (Right Shank, Left Shank, Right Thigh, Left Thigh) is a feasible solution (84.29% for NS, 81.68% for WS and 82.05% for DL). In particular, the latter statement will be fundamental for the future projects since it has already been planned to introduce IMU sensors even on ISS.

- Considering three subjects (among the 17 available of the IMU acquisition), who performed the highest number of repetitions, and studying them separately, the good accuracy results obtained (3 Female subjects, mean of accuracy for SVM

and CNN, respectively: 98.61% ± 2.41% and 95.83% ± 4.17% for NS; 98.61% ± 2.41% and 87.5 % ± 4.17% for WS; 91.67% ± 3.61% and 87.5% ± 10.83% for DL) highlight how building a customized classifier (suited for a single person) is a feasible pathway. This outcome is in agreement with the MARS-PRE project's aim of creating a customized biofeedback for single subjects.

- Inertial (e.g. coming from IMU) and simulated (related to OpenSim) data, gravity free, show visible differences and seem quite incompatible: currently, training ML algorithms with the former and testing with the latter (and viceversa) is not a reliable approach.

- Even simulated data require a much larger dataset (about 30 observations for each exercise type is very little, especially for the convolutional neural network).

The findings of this study have to be considered in the light of some limitations. First of all, the main drawback regards the number of subjects involved, which should be increased to conduct simulations and to reach better machine learning results in terms of accuracy. Indeed, data collected from 17 people is not enough, especially for CNN: in literature, it is well known how any type of neural network requires a huge amount of data for training. Even though the current project expected to collect more data, the instability caused by the Covid-19 virus strictly imposed limitations and did not allow to reach such purposes. However, we strongly believe that, enlarging the acceleration dataset, Convolutional Neural Networks will reach higher performances and will be very useful as biofeedback on ISS for the cosmonauts' workout. Furthermore, the solution considering the homolateral position of sensors is NOT a feasible option, since the asymmetry would be a limit and would imply feedbacks related only to a single body part (left or right) excluding the other one.

6.1 Future works

The first MOCAP system installed on board the ISS was ELITE-S2 (Ferrigno *et al.*, 2003), but recently NASA managed to install a new heir: the BTS-Smart system. The same device is also available in Politecnico di Milano and will allow to obtain useful information through the comparison of data obtained on ISS with the ones collected on ground. The current in-flight protocol should involve only a few markers placed on one body side. Of course, issues will arise during the process of wearing passive markers: just imagine all the concerns (even in terms of time) when putting on those markers in microgravity conditions! Some astronauts have already given their approval to conduct data collection during training in the next missions. These data will contribute to improve the weightlessness simulations, to conduct subject specific biomechanical analysis and to test the classifier with real 0G kinematics. Eventually, the final goals regard the introduction of IMU sensors to be used with ARED on ISS.

Moreover, the real time biofeedback could have a positive impact on the astronauts, psychologically speaking. It is well known how the space travelers does not perceive physical workout as a recreational daily moment and consequently they are often demotivated to train. The psychological aspect of motivation, which guarantees adherence to the pre-established training program, is essential especially talking about long-term missions. Perhaps, the real time coaching system, once implemented, will grant a more interactive experience for the cosmonauts.

In addition, at the very end of the thesis project we started working on a more realistic simulation integrating the ‘Catelli’ Model (Catelli et. al, 2018. Validation confirmed also in chapter 5.1) and the CAD of ARED already developed by Fregly et. al (2015). Since the old version of ARED had a script written in OpenSim 3.1 software, we adapted the HTML code for version 4.1. We also added the ‘Catelli’ features, but still some adjustments and improvements have to be made (Figure 76).
After these steps, it will be feasible to simulate in a more realistic way the cosmonauts’ workout related to new kinematic data acquired on ISS.

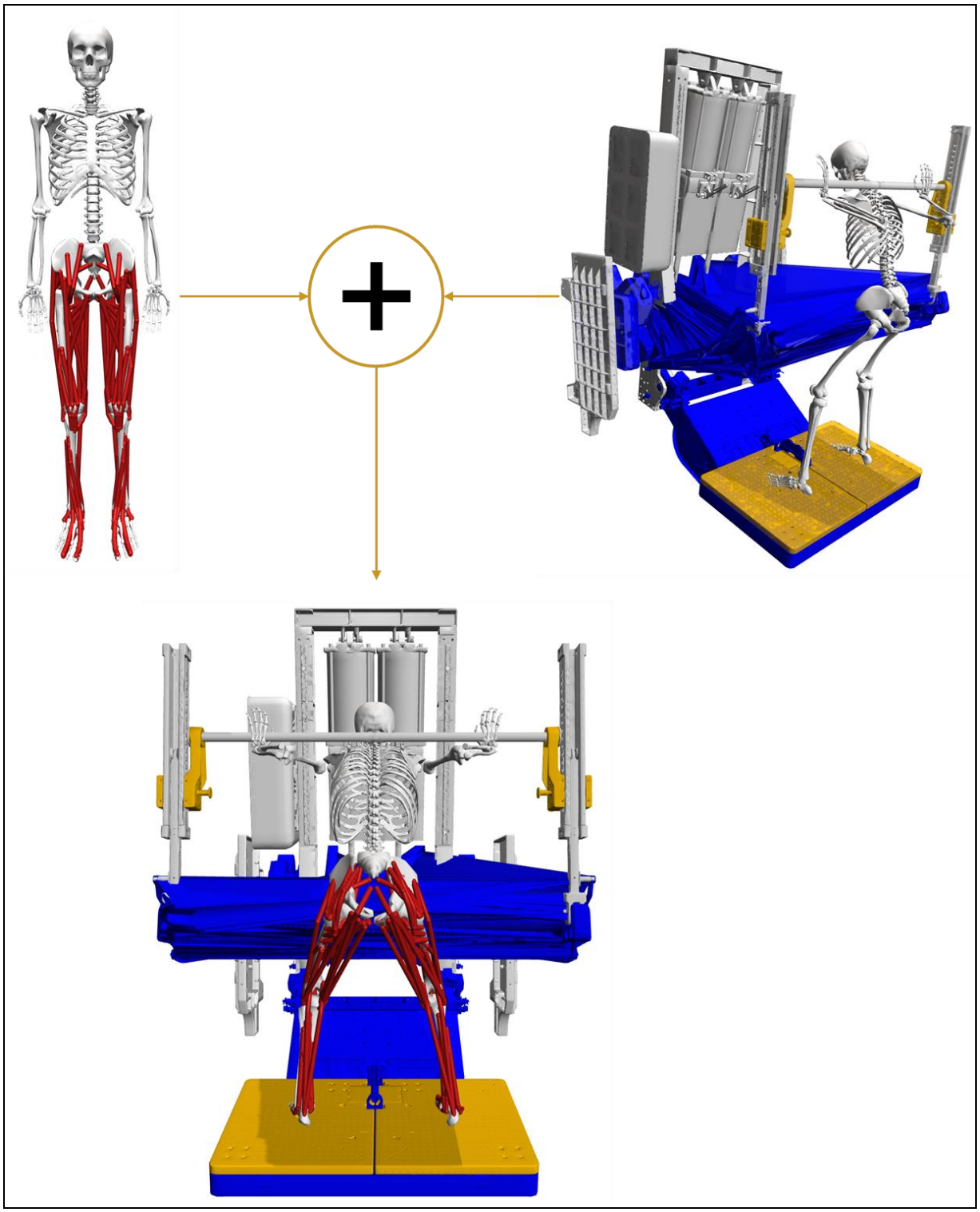


Figure 76: ‘Catelli’ model (upper left panel) and CAD of ARED (upper left panel) by Fregly et al. (2015). Our primitive version of their combination is shown in the lower part of the figure.

To the ones who will keep on working and will inherit the burden of this project, we feel like saying:

Houston …

we will always have problems,

but … hey…

the journey has just begun!!!

Year 2804: the thesis project's authors pretending to be astronauts. © All rights reserved.

BIBLIOGRAPHY

Avci O., Abdeljaber O., Kiranyaz S., Inman D., “Structural damage detection in real time: Implementation of 1D convolutional neural networks for SHM applications”. (2017)

Borrego M., Yadira Garcia Zaruba, James L. Broyan, Melissa K. McKinley and Shelley Baccus., “Exploration Toilet Integration Challenges on the International Space Station.”. (2019)

Capri M., Morsiani C., Santoro A., Moriggi M., Conte M., Martucci M. et al., "Recovery from 6-month spaceflight at the International Space Station: Muscle-related stress into a proinflammatory setting". (2019)

Catelli Danilo S., Mariska Wesseling, Ilse Jonkers & Mario Lamontagne, “A musculoskeletal model customized for squatting task”, Computer Methods in Biomechanics and Biomedical Engineering, DOI: 10.1080/10255842.2018.1523396. (2018)

De Boer M., Seynnes O., di Prampero P., Pišot R, Mekjavić I., Biolo G. et al., "Effect of 5 weeks horizontal bed rest on human muscle thickness and architecture of weight bearing and non-weight bearing muscles". (2008)

DeWitt John K., Fincke Renita S., Logan Rachel L., Guilliams Mark E., Ploutz-Snyder Lori L., "Load variation influences on joint work during squat exercise in reduced gravity". (2019)

Di Prampero Pietro E., Narici Marco V., "Muscles in microgravity: From fibres to human motion". (2003)

Ferchette A., ARED Photo/TV Reference Document. *ESA Standard Document*. (2017).

Ferrando AA, Lane HW, Stuart CA, Davis-Street J, Wolfe RR. “Prolonged bed rest decreases skeletal muscle and whole body protein synthesis”. Am J Physiol. (1996)

Ferretti G., Antonutto G., Denis C., Hoppeler H., Minetti A. E., Narici Marco V., Desplanches D., "Changes in electrically evoked skeletal muscle contractions during 17-day spaceflight and bed rest". doi: 10.1152/ajpendo.1996.270.4.E627. PMID: 8928769. (1997)

Ferrigno G., Pedrocchi A., Baroni G., Zolesi V., Bracciaferri F., Pedotti, A., “Elite S2 – A new instrument for multifactorial movement analysis on the international space station”. *54th International Astronautical Congress of the International Astronautical Federation, the International Academy of Astronautics, and the International Institute of Space Law. (*29 September - 3 October 2003).

Fregly B. J., Fregly C. D., Kim B. T., “Computational prediction of muscle moments during ARED squat exercise on the International Space Station”. Journal of Biomechanical Engineering 137, 121005. (2015)

Garrett-Bakelman F. E., Darshi M., Green S. J. et al., “The NASA Twins Study: A multi-omic, molecular, physiological, and behavioral analysis of a year-long human spaceflight”. *Science*. (2019)

Gastaldi Laura, Pastorelli Stefano Paolo, Muscolo Giovanni Gerardo, Musso Matteo, “Simulation HMI with OpenSim”. (2019)

Gemert T. Van., Syeda Hussaini, Hasti Narimanzadeh, “Real-time Human Activity classification using Convolutional Neural Networks with Raw Smartphone Sensor Data”. (2019)

Grant Schaffner, Eric J. Stetz, “Simulation of Squat Exercise Effectiveness Utilizing a Passive Resistive Exoskeleton in Zero Gravity”. (2016)

Hargens Alan R., Roshmi Bhattacharya, Suzanne M. Schneider, “Space physiology VI: exercise, artificial gravity, and countermeasure development for prolonged space flight”. European journal of applied physiology. (2012)

Haus J. M., Carrithers J. A., Carroll C. C., Tesch P. A., Trappe T. A., "Contractile and connective tissue protein content of human skeletal muscle: Effects of 35 and 90 days of simulated microgravity and exercise countermeasures". (2007)

Heer M., Paloski W. H., “Space motion sickness: incidence, etiology, and countermeasures”. (2006)

Jong In Han, Ho Seon Choi, Yoon Su Baek, “Simulation of a Lower Extremity Assistive Device for Resistance Training in a Microgravity Environment”. *Applied sciences MDPI.* (2020)

Kiranyaz S., Avci O., Abdeljaber O., Ince T., Gabbouj M., Inman D. J., “1D convolutional neural networks and applications - A survey”. (2019)

Lee J., Joo H., Lee J., Chee Y., “Automatic classification of squat posture using inertial sensors: Deep learning approach”. (2020)

Loehr J.A., Lee S. M. C., English K. L., Sibonga J., Smith S. M., Spiering B. A. et al., "Musculoskeletal adaptations to training with the advanced resistive exercise device". (2011)

Mansourifar H., Shi W., “Deep synthetic minority over-sampling technique”, Journal of Artificial Intelligence Research. (2002)

Menolotto Matteo, Komaris Dimitrios-Sokratis, Tedesco Salvatore, O’Flynn Brendan, Wals Michael, “Motion Capture Technology in Industrial Applications: A Systematic Review”. (2020)

Narici M V., De Boer MD., "Disuse of the musculo-skeletal system in space and on earth". (2011)

O'Reilly Martin, Duffin Joe, Ward Tomas, Caulfield Brian, “Mobile App to Streamline the Development of Wearable Sensor-Based Exercise Biofeedback Systems: System Development and Evaluation”. (2017)

O’Reilly M., Whelan DF, Ward TE, Delahunt E, Caulfield BM., “Classification of deadlift biomechanics with wearable inertial measurement units”. (2017)

O’Reilly M., Whelan D, Chanialidis C, Friel N, Delahunt E, Ward T, et al., “Evaluating squat performance with a single inertial measurement unit”. (2015)

Pavy-Le Traon A., Heer M., Narici M. V., Rittweger J., Vernikos J., "From space to Earth: Advances in human physiology from 20 years of bed rest studies (1986-2006)". (2007)

Petersen N., Jaekel P., Rosenberger A., Weber T., Scott J., Castrucci F. et al., "Exercise in space: The European Space Agency approach to in-flight exercise countermeasures for long-duration missions on ISS". (2016)

Prisk G., “Microgravity and the respiratory system”. The European respiratory journal. (2014)

Prisk G., “Pulmonary challenges of prolonged journeys to space: taking your lungs to the moon”. Med J (2019)

Rittweger J., Albracht K., Flück M., Ruoss S., Brocca L., Longa E. et al., "Sarcolab pilot study into skeletal muscle’s adaptation to longterm spaceflight". (2018)

Salanova M., Schiffl G., Püttmann B., Schoser G., Blottner D., "A method for determining Body Weight Replacement Load during Squat Exercise in Weightlessness". (2017)

Salanova M., Schiffl G., Püttmann B., Schoser G., Blottner D., "Molecular biomarkers monitoring human skeletal muscle fibers and microvasculature following long-term bed rest with and without countermeasures". (2008)

Shrestha A., Ji Dang. “Deep Learning-Based Real-Time Auto Classification Of Smartphone Measured Bridge Vibration Data”. (2020)

Vercellis Carlo, “Business Intelligence: Data Mining and Optimization for Decision Making “. (2009)

Vernice Nicholas A., Meydan Cem, Afshinnekoo Ebrahim, Mason Christopher E., “Long-term spaceflight and the cardiovascular system”, *Precision Clinical Medicine*, Volume 3, Issue 4, Pages 284–291, https://doi.org/10.1093/pcmedi/pbaa022 (2020)

Wee Chang An, Mohd Zamani Ngali, Zulhilmi Kaharuddin and Siti Badriah Khairu Razak, “Performance of Dual Depth Camera Motion Capture System for Athletes’ Biomechanics Analysis”. (2017)

X. Wang et al., “LRG1 promotes angiogenesis by modulating endothelial TGF-b signalling”. Nature 499, 306–311. (2013)

Zwart S. R. et al., “Genotype, B-vitamin status, and androgens affect spaceflight-induced ophthalmic changes”. FASEB J. 30, 141–148. (2016)

Zwart S. R. et al., “Vision changes after spaceflight are related to alterations in folate- and vitamin B-12-dependent one-carbon metabolism”. (2012)

https://www.esa.int/Space_in_Member_States/Italy/La_Stazione_Spaziale_Internazionale_l_Europa_e_l_Italia [21/08/2020]

https://ntrs.nasa.gov/citations/20170004124 [21/08/2020]

https://www1.grc.nasa.gov/space/human-research-program/explore/computational-modeling/musculoskeletal-modeling/musculoskeletal/ [10/09/2020]

https://www.nasa.gov/analogs/envihab/bed-rest-faqs [11/09/2020]

https://simtk.org/frs/?group_id=1043 [25/09/2020]

https://simtk-confluence.stanford.edu:8443/display/OpenSim/OpenSim+Documentation [10/10/2020]

https://github.com/opensim-org/opensim-core [10/10/2020]

https://simtk.org/api_docs/opensim/api_docs/index.html [15/10/2020]

https://simtk-confluence.stanford.edu/display/OpenSim/User's+Guide [15/10/2020]

https://it.mathworks.com/help/signal/ug/signal-smoothing.html?prodcode=SG&language=en [5/11/2020]

https://machinelearningmastery.com/cnn-models-for-human-activity-recognition-time-series-classification/ [28/11/2020]

https://github.com/laxmimerit/Human-Activity-Recognition-Using-Accelerometer-Data-and-CNN [30/11/2020]

https://github.com/bharatm11/1D_CNN_Human_activity_recognition [30/11/2020]

https://github.com/jeandeducla/ML-Time-Series [2/12/2020]

https://machinelearningmastery.com/evaluate-machine-learning-algorithms-for-human-activity-recognition/ [5/12/2020]

https://github.com/fandulu/DD-Net [6/12/2020]

https://towardsdatascience.com/what-is-xgboost-and-how-to-optimize-it-d3c24e0e41b4 [10/12/2020]

bbhttps://it.mathworks.com/help/signal/ref/findpeaks.html?searchHighlight=find%20peak&s_tid=srchtitle [12/12/2020]

https://www.datacamp.com/community/tutorials/principal-component-analysis-in-python [14/12/2020]

https://blogs.nasa.gov/northropgrumman/ [5/01/2021]

https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/ [06/01/2021]

https://www.nasa.gov/audience/forstudents/5-8/features/nasa-knows/what-is-the-iss-58.html [15/01/2021]

https://appel.nasa.gov/2012/05/01/the-nasa-digital-astronaut-project/ [15/01/2021]

https://machinelearningmastery.com/save-load-keras-deep-learning-models/ [29/01/2021]

APPENDIX A

Input and output OpenSim files explanation

Scale Tool files

- *subject01_model.osim* is the biomechanical model which will be used for the simulation;
- *subject01_static.trc* contains coordinates of experimental marker for a static trial. The latter is usually several seconds of data with the subject posed in a known static position. A segment of a regular motion file can be used as a static trial if desired.
- *ScaleMarkerSet.xml* contains coordinates of virtual marker to place on the model, which reproduce exactly the disposition of the experimental markers put on the subject during the data collection;
- *subject01_Setup_Scale.xml* is a file containing all the setting information for the Scale Tool.
- *subject01_scaled_model.osim* is the output of the Tool, therefore it is the musculoskeletal model scaled to the dimensions of the subject.

Inverse Kinematic Tool files

- *subject01_simbody.osim* is the subject-specific OpenSim model generated by scaling a generic model with the Scale Tool, along with an associated marker set containing adjusted virtual markers;
- *subject01_walk1.trc* contains experimental marker trajectories of a trial obtained from a motion capture system, along with the time range of interest;
- *subjec01_Setup_IK.xml* is a file containing all the setting information for the Tool;
- *subject01_walk1_ik.osim* is the output of the Tool, therefore it is a motion file containing the generalized coordinate trajectories (joint angles and/or translations) computed by IK.

Inverse Dynamic Tool files

- *subject01_simbody.osim* is the subject-specific OpenSim model generated by scaling a generic model with the Scale Tool, along with an associated marker set containing adjusted virtual markers. The model must include inertial parameters;
- *subject01_walk1_grf.xml* is a file containing all external load data (i.e., ground reaction forces, moments, and center of pressure location). This file includes also the name of each force as well the names of the bodies to which they are applied;
- *subject01_walk1.mot* is the result of the IK, therefore a motion file containing time histories of joint angles;
- *subjec01_Setup_InverseDynamics.xml* is a file containing all the setting information for the Tool;
- *subject01_walk1_InverseDynamics_force.sto* is the output of the Tool, therefore it is a storage file containing the time histories of the net joint moments.

Residual Reduction Algorithm Tool files

- *subject01_simbody.osim* is the A subject-specific OpenSim model generated by scaling a generic model with the Scale Tool or by other means, along with an associated marker set containing adjusted virtual markers. The model must include inertial parameters;
- *subject01_walk1_grf.xml* is a file containing all external load data data (i.e., ground reaction forces, moments, and center of pressure location). This file includes also the name of each force as well the names of the bodies to which they are applied;
- *subject01_walk1.mot* is the result of the IK, therefore a motion file containing time histories of joint angles;
- *subjec01_Setup_RRA.xml* is a file containing all the setting information for the Tool;
    - *gait2354_RRA_Actuators.xml* is a file specifying the residual and reserve actuators to be applied and their parameters, such as maximum/minimum force.

- *gait2354_RRA_Tasks.xml* is a tracking file that specifies which coordinates to track and the corresponding weights, used to determine how well a joint angle will tracks the specified joint angle from IK.
- *subject01_walk1_RRA_Kinematic_q.sto* is the output of the Tool, therefore it is a storage file containing the time histories of the net joint moments.
- *Subject01_simbody_adjusted.osim* is a model with adjusted mass properties.

Compute Muscle Control Tool files

- *subject01_simbody_adjusted.osim* is the subject specific model with adjusted mass properties obtained with RRA;
- *subject01_walk1_grf.xml* is a file containing all external load data(i.e., ground reaction forces, moments, and center of pressure location). This file includes also the name of each force as well the names of the bodies to which they are applied;
- *subject01_walk1_RRA_Kinematic_q.sto* is the output of RRA tool, therefore it is a storage file containing the time histories of the net joint moments.
- *subjec01_Setup_CMC.xml* is a file containing all the setting information for the Tool;
- *gait2354_CMC_Actuators.xml* is a file specifying the residual and reserve actuators to be applied and their parameters, such as maximum/minimum force.
- *gait2354_CMC_Tasks.xml* is a tracking file that specifies which coordinates to track and the corresponding weights, used to determine how well a joint angle will tracks the specified joint angle from IK.
- *gait2354_CMC_ControlConstraints.xml* is a file which contains limits on model actuators, including muscles, reserve and residual actuators. Control constraint file specifies maximum and minimum excitation for each actuator.
- *subject01_simbody_controls.xml* is the output of the tool containing the excitations to individual muscles as well controls for any residual and reserve actuators.
- *subject01_simbody_forces.sto* is a file with muscle forces and reserve or residual forces and torques.

APPENDIX B

IMU DATA - NORMAL SQUAT

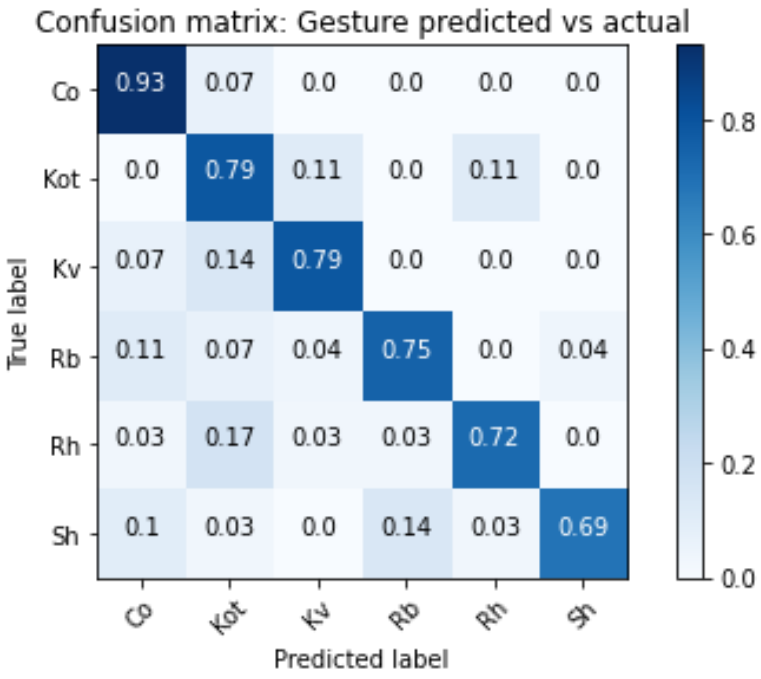
K-nearest neighbor (KNN) results

Parameters:
k = 4

Test set report

	precision	recall	f1-score	support
CO	0.75	0.93	0.83	29
Kot	0.61	0.79	0.69	28
Kv	0.82	0.79	0.81	29
Rb	0.81	0.75	0.78	28
Rh	0.84	0.72	0.78	29
Sh	0.95	0.69	0.80	29
accuracy			0.78	172
macro avg	0.80	0.78	0.78	172
weighted avg	0.80	0.78	0.78	172

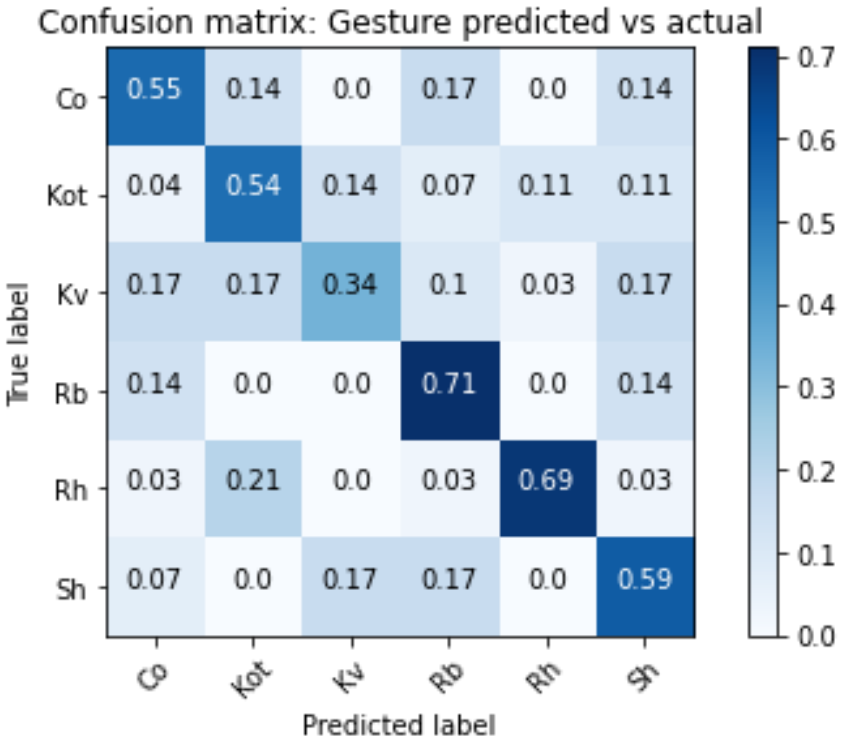
Accuracy: 77.91%



Decision tree results

	precision	recall	f1-score	support
CO	0.55	0.55	0.55	29
Kot	0.50	0.54	0.52	28
Kv	0.53	0.34	0.42	29
Rb	0.56	0.71	0.63	28
Rh	0.83	0.69	0.75	29
Sh	0.50	0.59	0.54	29
accuracy			0.57	172
macro avg	0.58	0.57	0.57	172
weighted avg	0.58	0.57	0.57	172

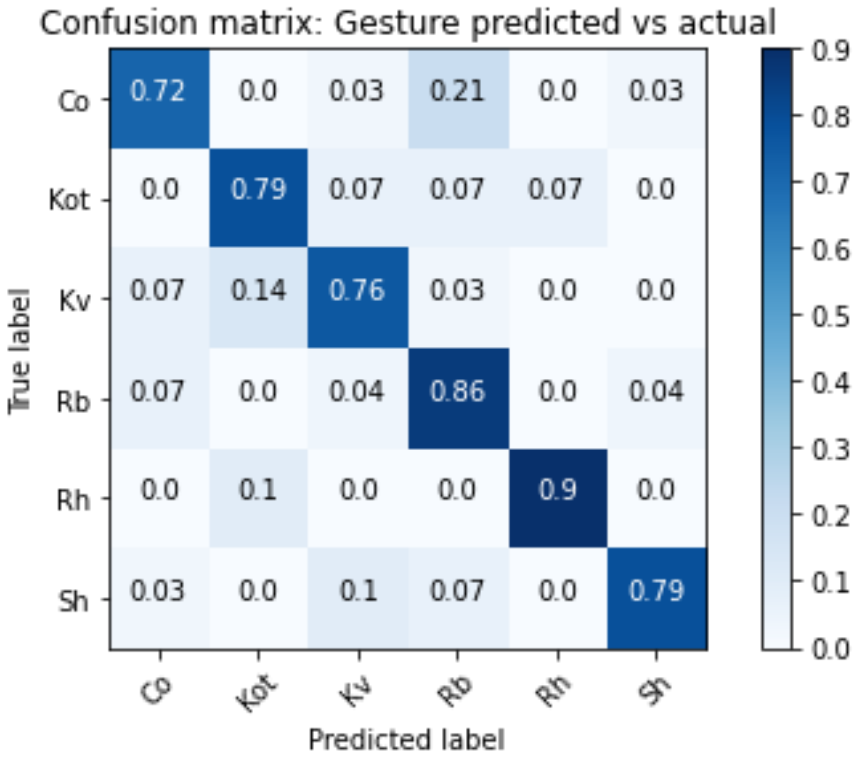
Accuracy: 56.98%



Extreme Gradient Boosting (XGBoost) results

	precision	recall	f1-score	support
CO	0.81	0.72	0.76	29
Kot	0.76	0.79	0.77	28
Kv	0.76	0.76	0.76	29
Rb	0.69	0.86	0.76	28
Rh	0.93	0.90	0.91	29
Sh	0.92	0.79	0.85	29
accuracy			0.80	172
macro avg	0.81	0.80	0.80	172
weighted avg	0.81	0.80	0.80	172

Accuracy: 80.23%

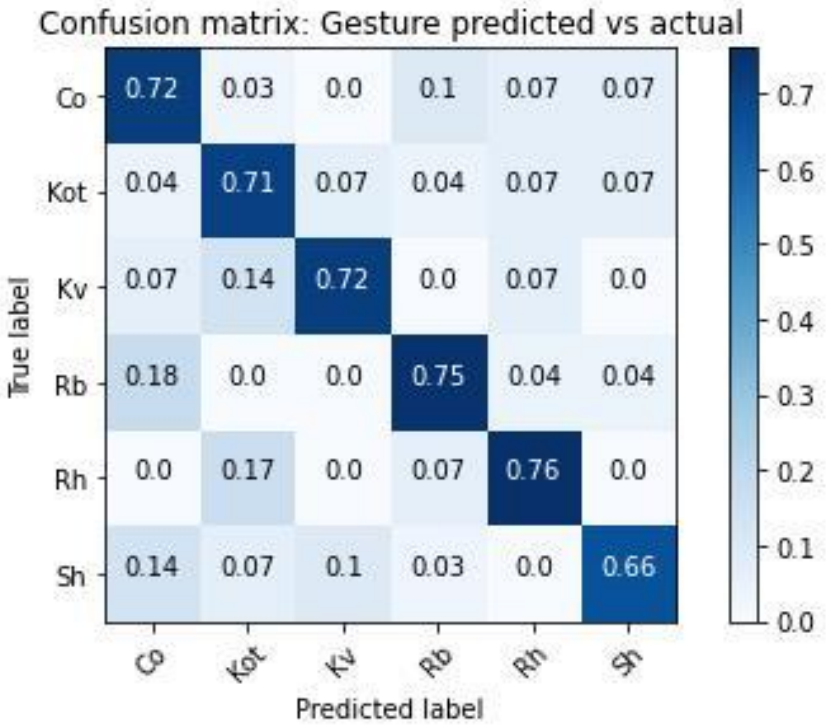


Multi-layer perceptron (MLP) results

Test set report

	precision	recall	f1-score	support
Co	0.64	0.72	0.68	29
Kot	0.62	0.71	0.67	28
Kv	0.81	0.72	0.76	29
Rb	0.75	0.75	0.75	28
Rh	0.76	0.76	0.76	29
Sh	0.79	0.66	0.72	29
accuracy			0.72	172
macro avg	0.73	0.72	0.72	172
weighted avg	0.73	0.72	0.72	172

Accuracy: 72.09 %



IMU DATA - WIDE SQUAT

K-nearest neighbor (KNN) results

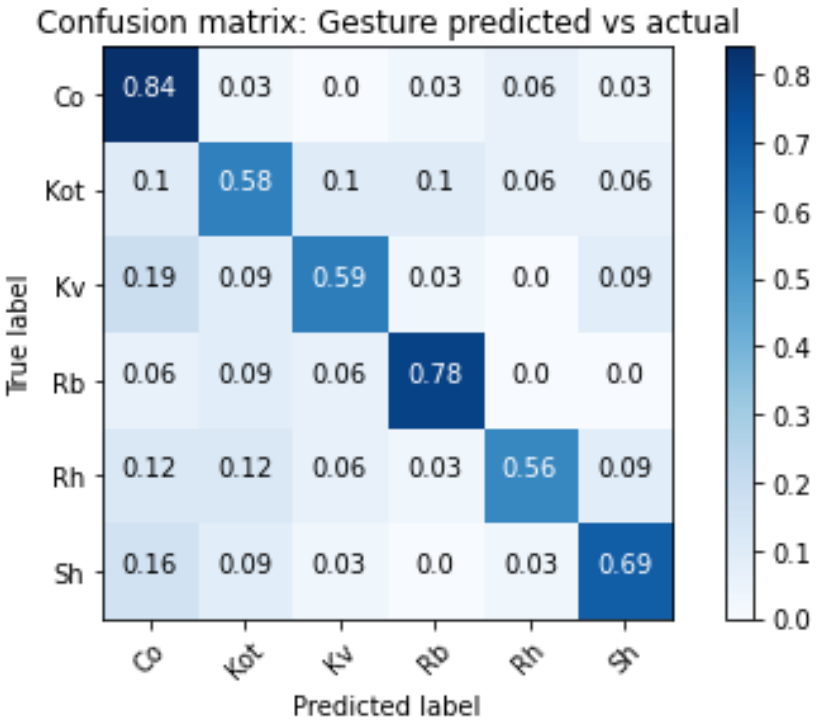
```
Parameters:
k = 3


Test set report
              precision    recall  f1-score   support

          CO       0.57      0.84      0.68        32
         Kot       0.56      0.58      0.57        31
          Kv       0.70      0.59      0.64        32
          Rb       0.81      0.78      0.79        32
          Rh       0.78      0.56      0.65        32
          Sh       0.71      0.69      0.70        32

    accuracy                           0.68       191
   macro avg       0.69      0.67      0.67       191
weighted avg       0.69      0.68      0.67       191
```

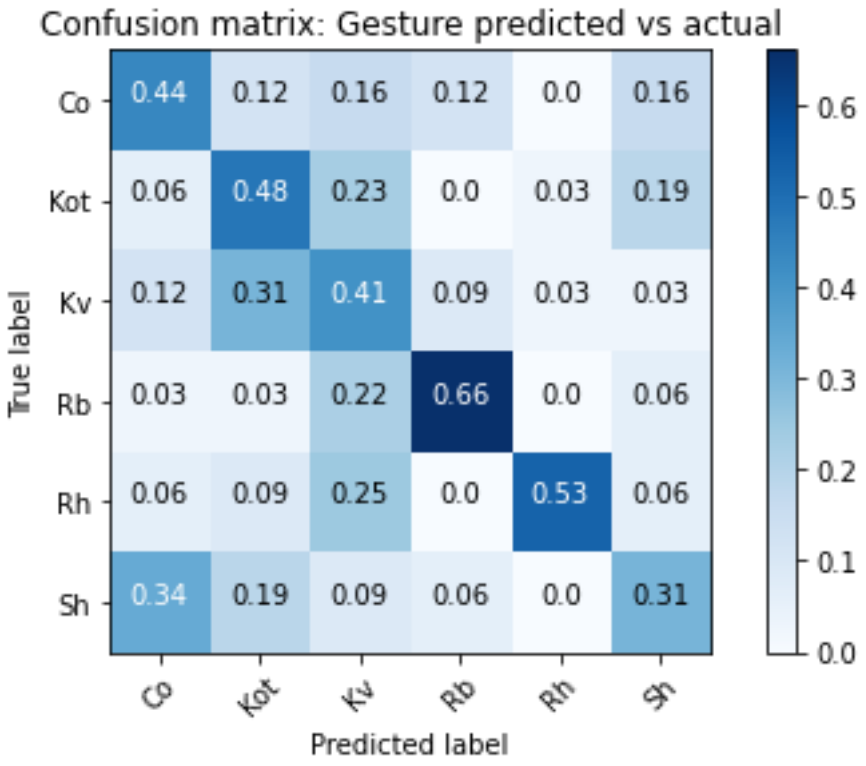
Accuracy: 67.54%



Decision tree results

	precision	recall	f1-score	support
CO	0.41	0.44	0.42	32
Kot	0.38	0.48	0.43	31
Kv	0.30	0.41	0.35	32
Rb	0.70	0.66	0.68	32
Rh	0.89	0.53	0.67	32
Sh	0.38	0.31	0.34	32
accuracy			0.47	191
macro avg	0.51	0.47	0.48	191
weighted avg	0.51	0.47	0.48	191

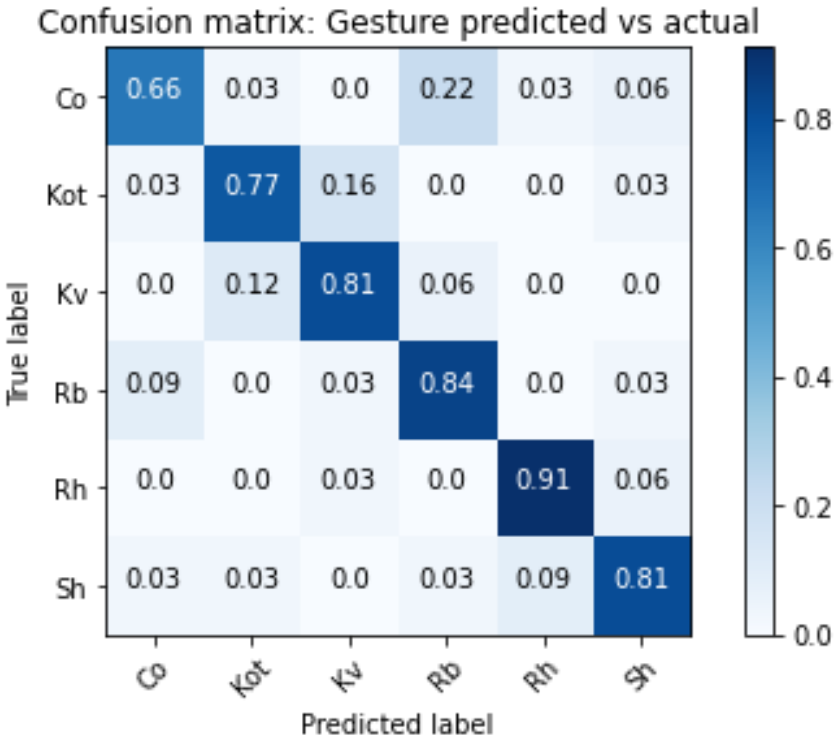
Accuracy: 47.12%



Extreme Gradient Boosting (XGBoost) results

	precision	recall	f1-score	support
CO	0.81	0.66	0.72	32
Kot	0.80	0.77	0.79	31
Kv	0.79	0.81	0.80	32
Rb	0.73	0.84	0.78	32
Rh	0.88	0.91	0.89	32
Sh	0.81	0.81	0.81	32
accuracy			0.80	191
macro avg	0.80	0.80	0.80	191
weighted avg	0.80	0.80	0.80	191

Accuracy: 80.10%

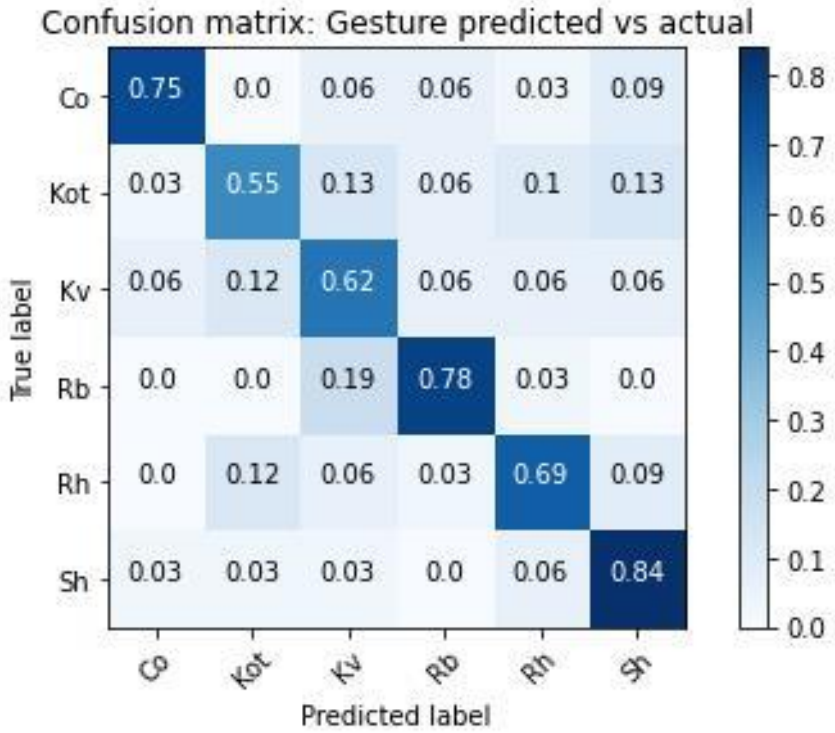


Multi-layer perceptron (MLP) results

Test set report

	precision	recall	f1-score	support
Co	0.86	0.75	0.80	32
Kot	0.65	0.55	0.60	31
Kv	0.57	0.62	0.60	32
Rb	0.78	0.78	0.78	32
Rh	0.71	0.69	0.70	32
Sh	0.69	0.84	0.76	32
accuracy			0.71	191
macro avg	0.71	0.71	0.71	191
weighted avg	0.71	0.71	0.71	191

Accuracy: 70.68%



IMU DATA - DEADLIFT

K-nearest neighbor (KNN) results

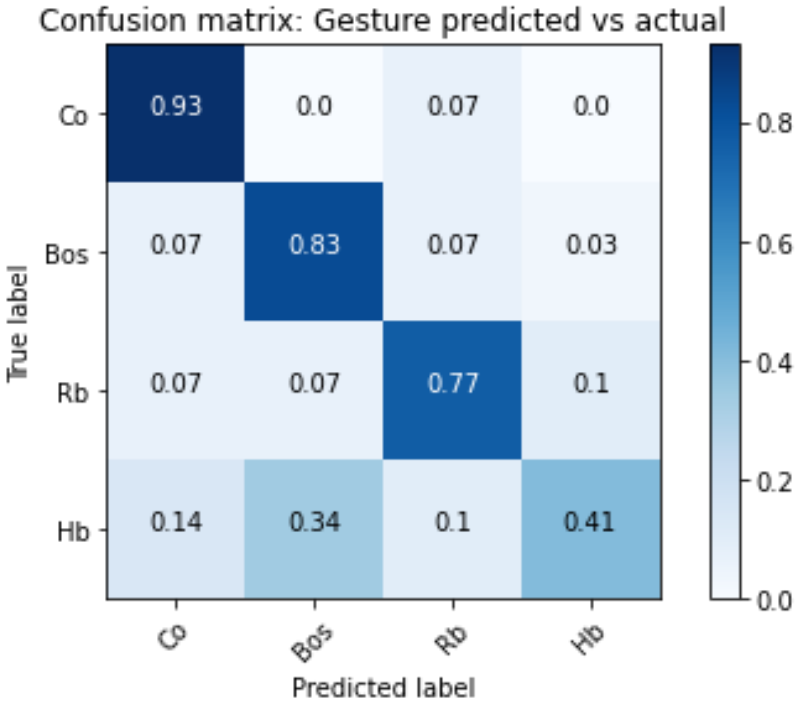
```
Parameters:
k = 4


Test set report
              precision    recall  f1-score   support

          Co       0.77      0.93      0.84        29
         Bos       0.67      0.83      0.74        29
          Rb       0.77      0.77      0.77        30
          Hb       0.75      0.41      0.53        29

    accuracy                           0.74       117
   macro avg       0.74      0.73      0.72       117
weighted avg       0.74      0.74      0.72       117
```

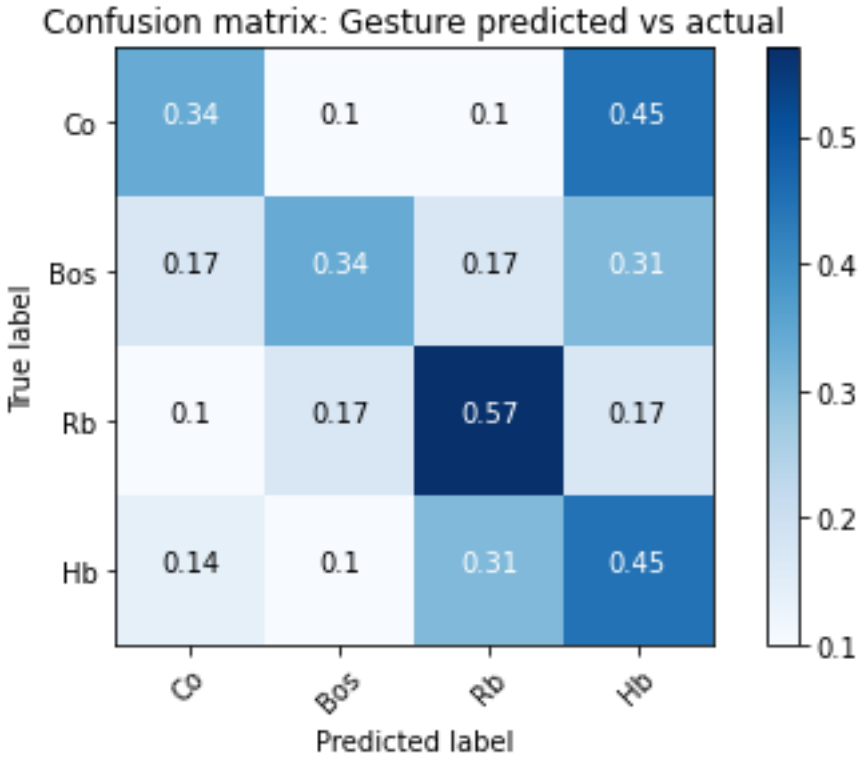
Accuracy: 73.50%



Decision tree

	precision	recall	f1-score	support
Co	0.45	0.34	0.39	29
Bos	0.48	0.34	0.40	29
Rb	0.50	0.57	0.53	30
Hb	0.33	0.45	0.38	29
accuracy			0.43	117
macro avg	0.44	0.43	0.43	117
weighted avg	0.44	0.43	0.43	117

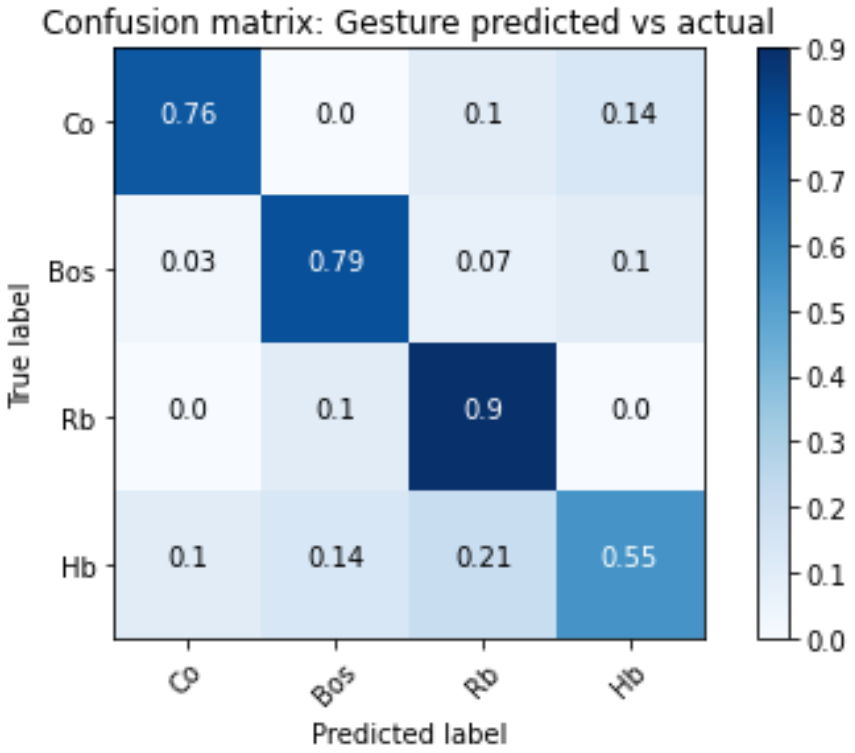
Accuracy: 42.74%



Extreme Gradient Boosting (XGBoost) results

	precision	recall	f1-score	support
Co	0.85	0.76	0.80	29
Bos	0.77	0.79	0.78	29
Rb	0.71	0.90	0.79	30
Hb	0.70	0.55	0.62	29
accuracy			0.75	117
macro avg	0.75	0.75	0.75	117
weighted avg	0.75	0.75	0.75	117

Accuracy: 75.21%

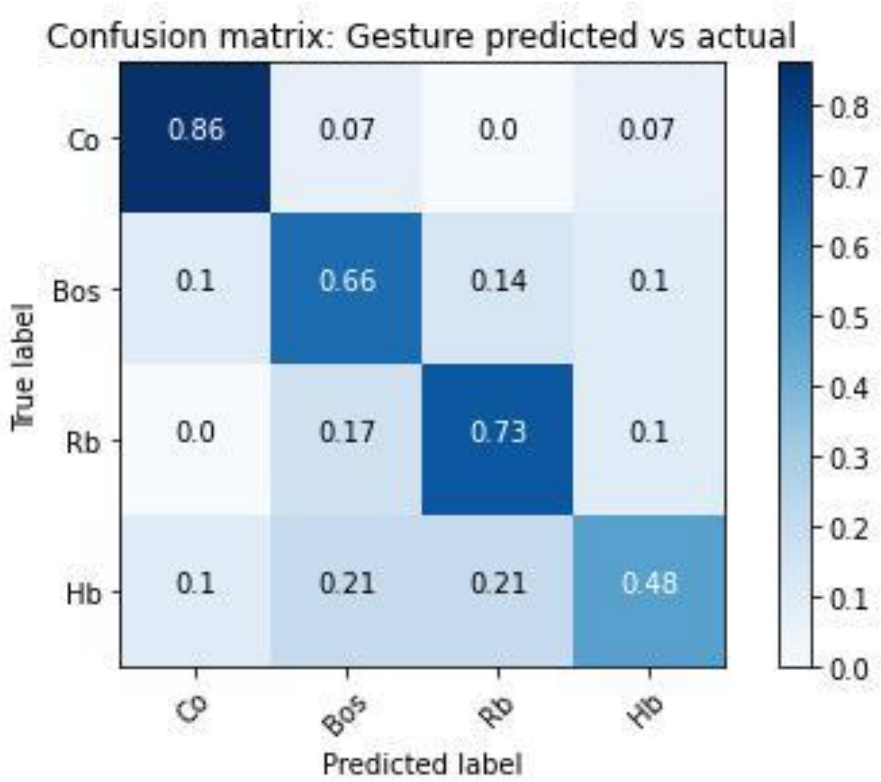


Multi-layer perceptron (MLP) results

Test set report

	precision	recall	f1-score	support
Co	0.81	0.86	0.83	29
Bos	0.59	0.66	0.62	29
Rb	0.69	0.73	0.71	30
Hb	0.64	0.48	0.55	29
accuracy			0.68	117
macro avg	0.68	0.68	0.68	117
weighted avg	0.68	0.68	0.68	117

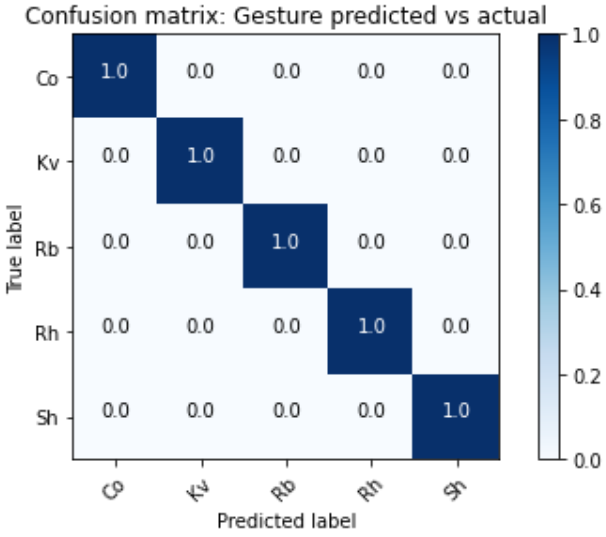
Accuracy: 68.37%



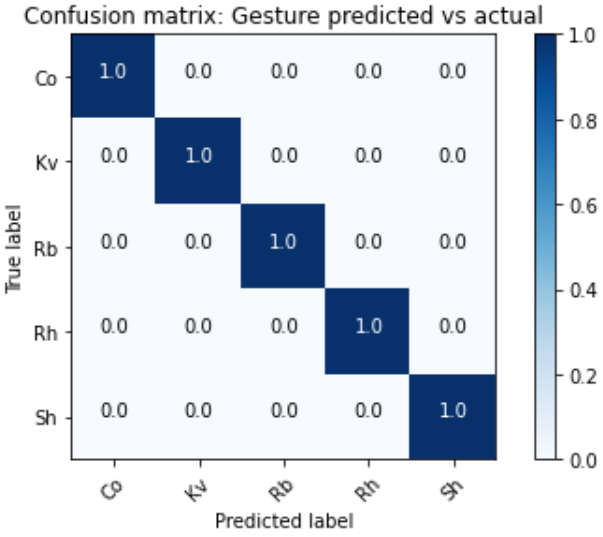
CUSTOMIZED CLASSIFIER FOR SINGLE SUBJECT

SUBJECT 06 – NORMAL SQUAT

Support Vector Machines

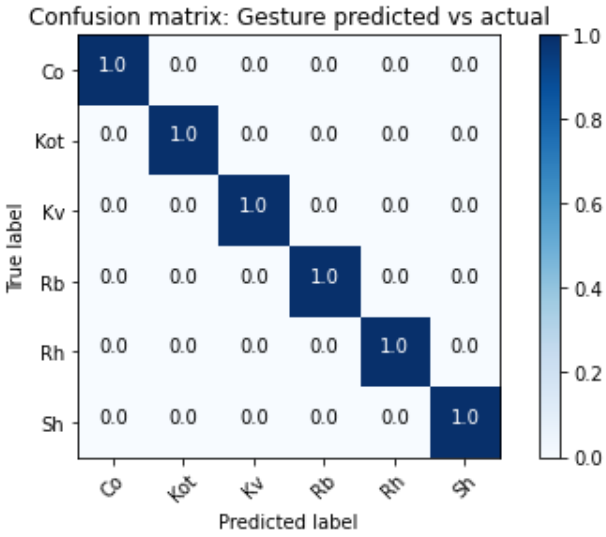


Convolutional Neural Network

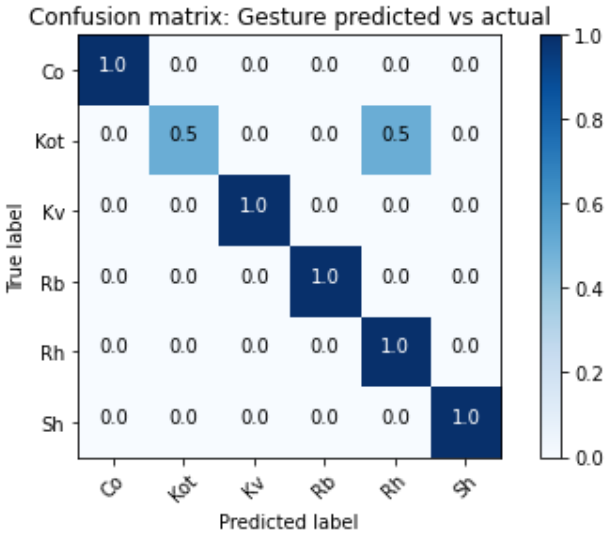


SUBJECT 06 – WIDE SQUAT

Support Vector Machines

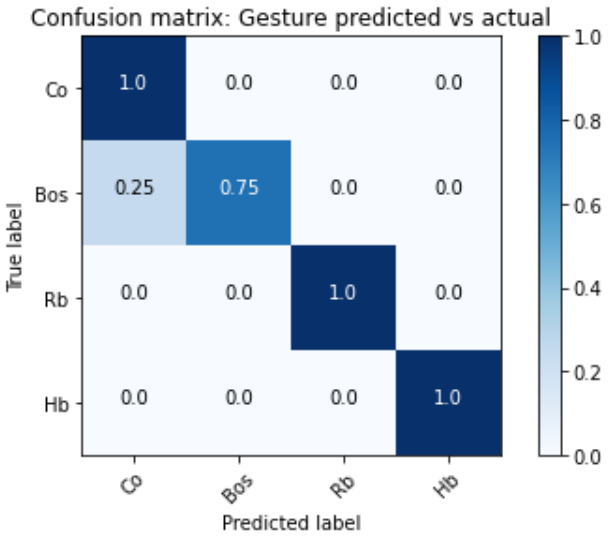


Convolutional Neural Network

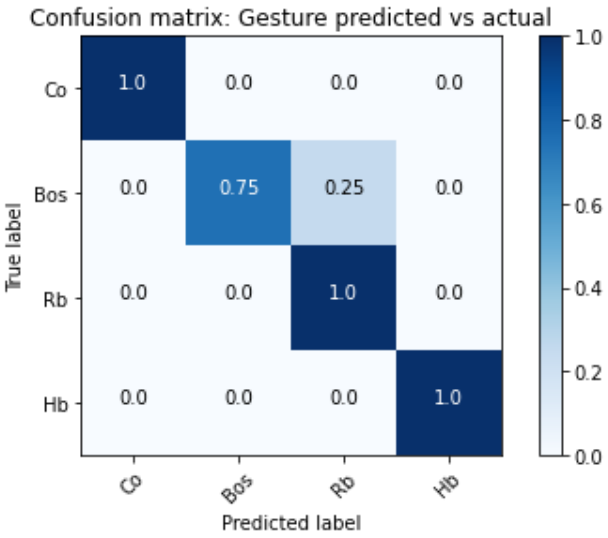


SUBJECT 06 – DEADLIFT

Support Vector Machines

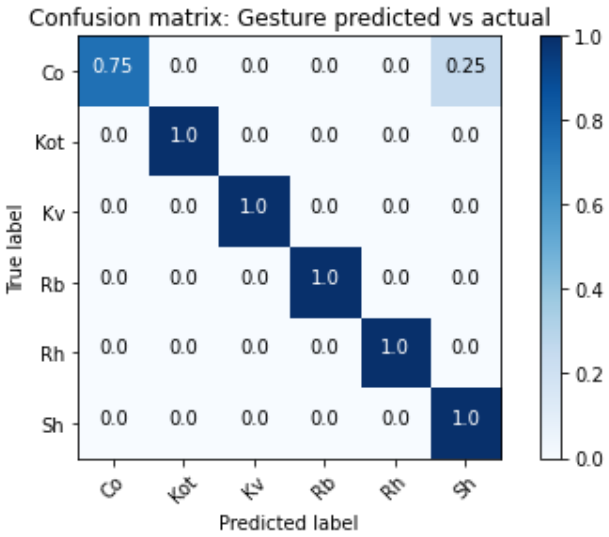


Convolutional Neural Network

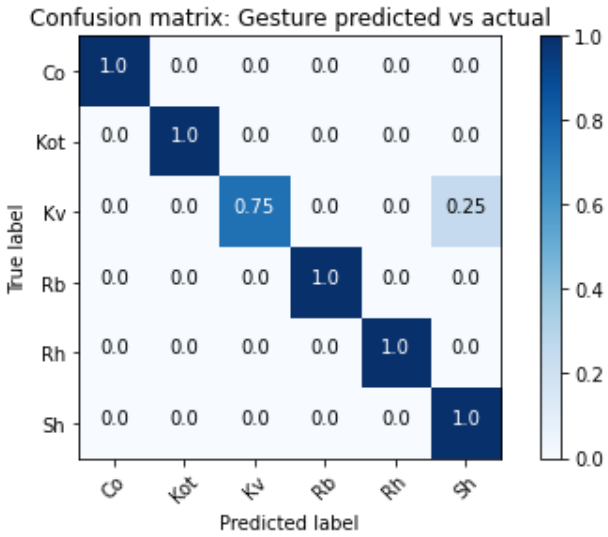


SUBJECT 08 – NORMAL SQUAT

Support Vector Machines

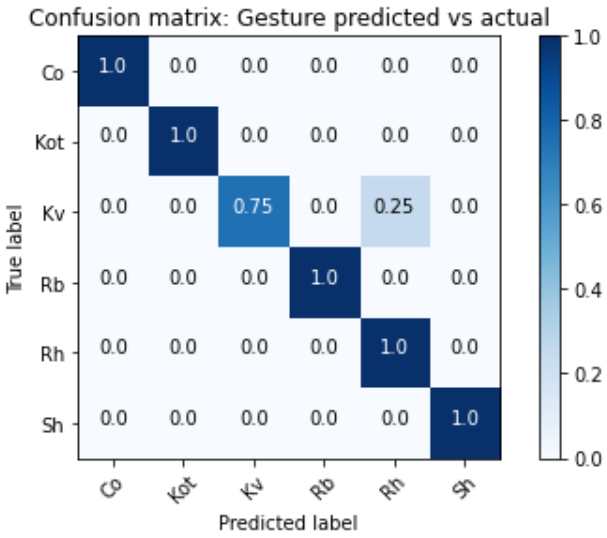


Convolutional Neural Network

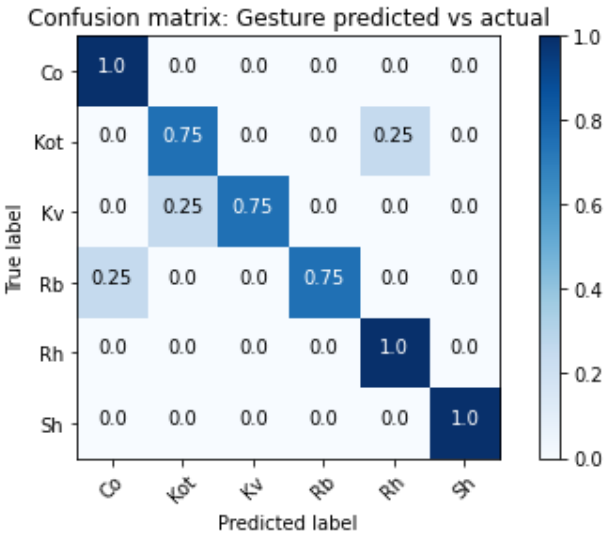


SUBJECT 08 – WIDE SQUAT

Support Vector Machines

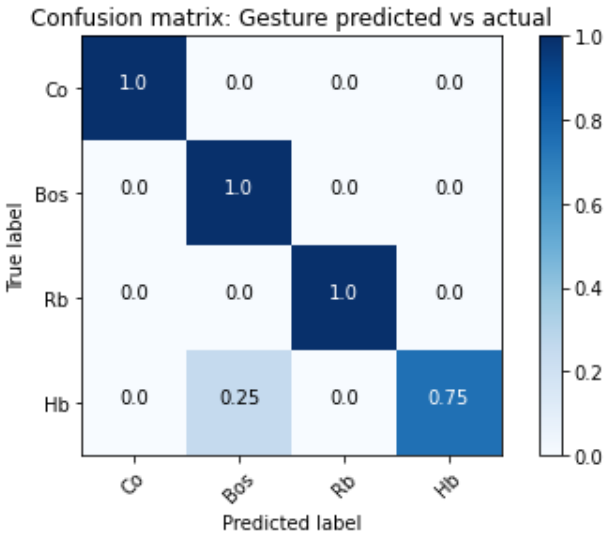


Convolutional Neural Network

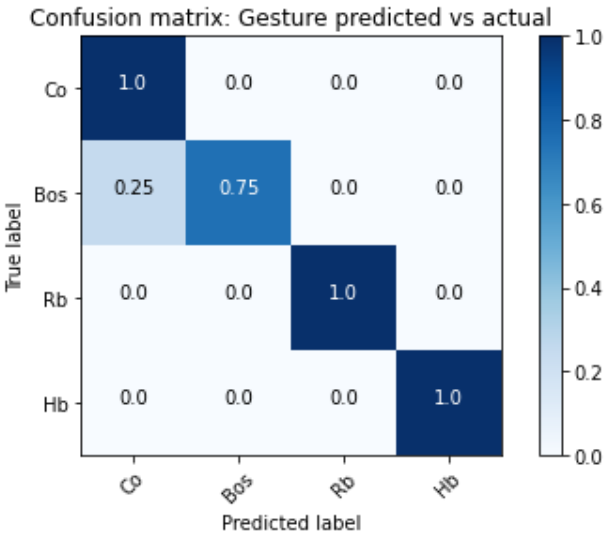


SUBJECT 08 – DEADLIFT

Support Vector Machines

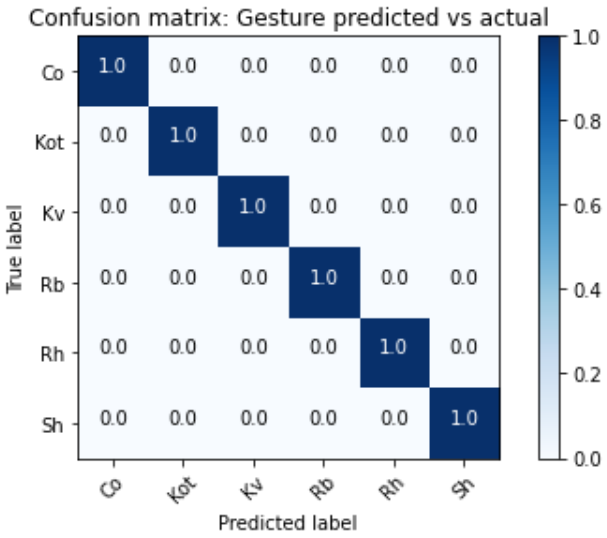


Convolutional Neural Network

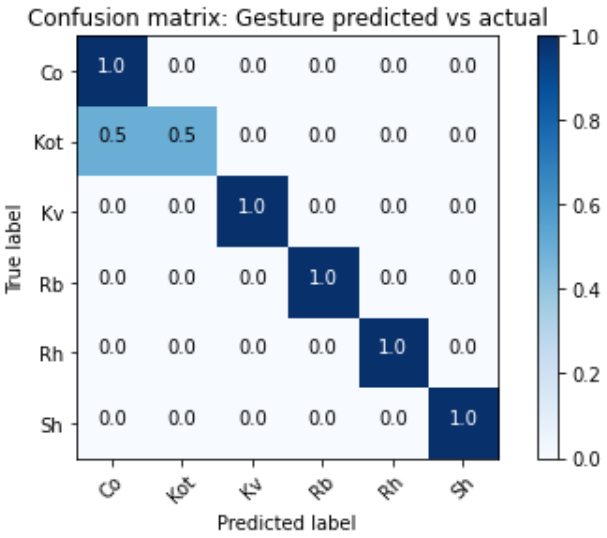


SUBJECT 11 – NORMAL SQUAT

Support Vector Machines

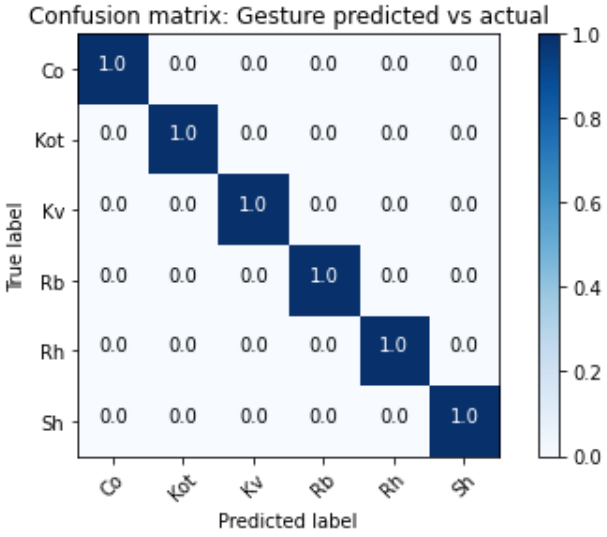


Convolutional Neural Network

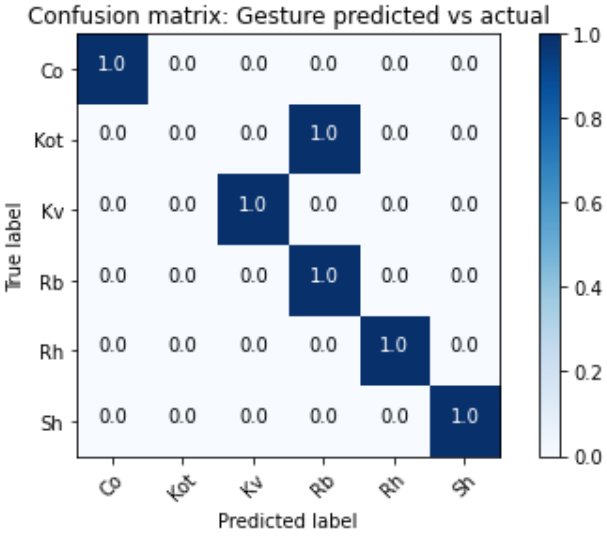


SUBJECT 11 – WIDE SQUAT

Support Vector Machines

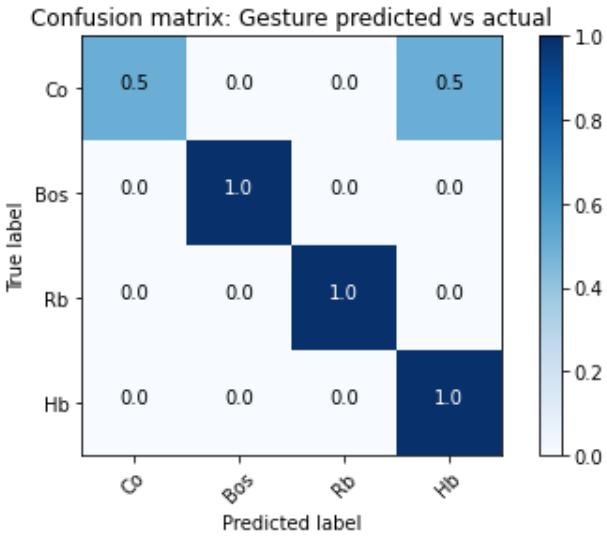


Convolutional Neural Network

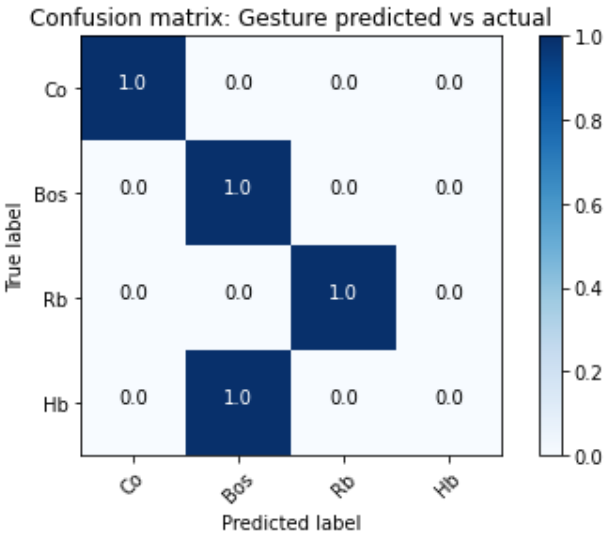


SUBJECT 11 – DEADLIFT

Support Vector Machines



Convolutional Neural Network



A summary of the results, in terms of accuracy, can be summarized in the followin table:

	S06		S08		S11	
	CNN	SVM	CNN	SVM	CNN	SVM
NS	100.00%	100.00%	95.83%	95.83%	91.67%	100.00%
WS	91.67%	100.00%	87.50%	95.83%	83.33%	100.00%
DL	93.75%	93.75%	93.75%	93.75%	75.00%	87.50%

SIMULATED DATA

This section regards the classification of simulated data (i.e. those coming from OpenSim software) starting from a classifier trained with real data. Below, tables referred to SVM and CNN approaches are reported. The first row of each table is the target variable (we know exactly how the exercise have been performed according to the protocol of correct/incorrect executions explained in chapter 4.2). The second row, instead, denotes the classification predicted by the ML algorithm. The columns highlighted in orange are the only ones predicted in a correct way. We clearly notice how neither SVM nor CNN are able to obtain discrete/good results.

REMINDER: A target array was created to define the correct class of each entry, labeled with {0, 1, 2, 3, 4, 5} when considering the Squat exercises (Normal Squat and Wide Squat), where 0 corresponded to a correct exercise (CO), 1 to Knees Over Toes (KOT), 2 to Valgus Knees (Kv), 3 to Rounded Back (Rb), 4 to Raised Heels (Rh), 5 to Shallow (Sh). When considering deadlift instead, the labels are {0, 1, 2, 3}, where 0 corresponded to a correct exercise (CO), 1 to Bar Over Shoulder (BOS), 2 to Rounded back (Rb), 3 to Hyperextended back (Hb).

Support Vector Machines (SVM) prediction results

Target NS	0	0	0	0	3	3	1	1	2	2	4	4	5	5	0	0	0	0	3	3	3	3	1	1	1	1	2	2	2	2	4	4	4	4	5	5	5
Predicted NS	1	4	4	0	4	3	1	0	3	2	4	1	1	5	5	5	2	5	1	3	5	3	3	3	3	1	3	3	1	1	5	2	0	2	2	4	3	4

Target WS	0	0	0	0	3	3	1	1	1	2	2	0	0	0	0	3	3	3	3	1	1	1	1	2	2	2	2
Predicted WS	5	5	5	2	1	5	5	5	0	3	5	2	0	3	5	4	1	4	2	1	2	2	2	2	1	2	4

Target DL	0	0	0	2	2	1	1	3	3	0	0	0	0	2	2	2	2	1	1	1	1	3	3	3	3
Predicted DL	3	3	1	0	3	2	2	0	3	2	1	3	1	2	2	2	2	1	3	3	2	1	1	0	1

Convolutional Neural Network (CNN) prediction results

Target NS	0	0	0	0	3	3	1	1	2	2	4	4	5	5	0	0	0	0	3	3	3	3	1	1	1	1	2	2	2	2	4	4	4	4	5	5	5
Predicted NS	0	4	0	0	0	0	4	4	1	0	0	0	0	0	0	0	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Target WS	0	0	0	0	3	3	1	1	1	2	2	0	0	0	0	3	3	3	3	1	1	1	1	2	2	2	2
Predicted WS	4	4	4	4	0	0	0	0	0	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Target DL	0	0	0	2	2	1	1	3	3	0	0	0	0	2	2	2	2	1	1	1	1	3	3	3	3
Predicted DL	3	3	3	3	3	3	3	3	3	3	1	3	3	3	0	0	0	1	0	0	3	1	3	2	0