MASTER THESIS

You get the best of both worlds!

sEMG to Knee Torque mapping using (Hybrid) Neuromusculoskeletal Modelling and Machine Learning

Marijke Zondag BSc.

FACULTY OF ELECTRICAL ENGINEERING, MATHEMATICS AND COMPUTER SCIENCE
BIOMEDICAL SIGNALS AND SYSTEMS

EXAMINATION COMMITTEE
prof. dr. J.H. Buurke (Chair)
ir. R.V. Schulte
dr. E.C. Prinsen
dr. G.V. Durandau

24-03-2021

Roessingh Research and Development

UNIVERSITY OF TWENTE.

I. Preface

Dear reader,

I am writing this preface with both a smile on my face and tears in my eyes. For six years, I have enjoyed being a student, having made friends for life and creating a home for myself in Enschede. This report confirms the end of what has been an amazing and life-changing time.

Since July 2020, I have been working on my thesis, of which you have the result in front of you. This research was my perfect ending of a fun and educational Master's education. My knowledge of biomechanics, signal analysis and data science was used directly into brand-new modelling frameworks. I was also very fortunate to have been able to conduct subject measurements for my own dataset, despite of having to deal with the COVID-19 measures. It was very pleasant and educational and I was lucky to be able to employ my social skills in my thesis research as well!

My thesis was carried out within the MyLeg project at Roessingh Research and Development (RRD). I am grateful that I got to be part of the MyLeg team. Everyone was kind, interested in each other's work and really helpful. I always felt like an equal member of the team and liked that I had an important role within the project. I want to thank Erik, Robert, Eline and also Kaz, Sibren and Ali for the fun weekly meetings and their contributions to my thesis. Special thanks go out to Erik; your practical view on my research was much appreciated. Engineers (or almost-engineers) tend to get lost in technical details and sometimes need to be reminded about the actual application. I also want a special shout-out to Robert. Not only did you guide me through my thesis research, but you also prepared me for the big scary world after. I feel like I have grown more confident about my knowledge and skills and this is much thanks to you!

Furthermore, I want to thank Jaap and Guillaume for their input and supervision during my thesis. I really appreciate your enthusiasm for my project, which added to my own motivation to conduct my research. Your feedback has definitely taken the quality of my thesis to the next level.

I believe that I would not have been able to finish my thesis the way I did now, if I did not have the amazing support of my family. And not only the "Zondag" family, but also the "BIEM!" family: my amazing roommates. You all forced me (with love) to sometimes let go of my work and take time for myself. I want to thank my friends as well. Your advice, love and kindness helped me endure all the highs and lows that were thrown my way.

I wish you all a pleasant reading of the report I am so very proud of!

Marijke Zondag

II. SAMENVATTING

[1]Proportionele aansturing met behulp van oppervlakte elektromyografie (sEMG) biedt meer intuïtieve aansturing van een transfemorale prothese. sEMG is echter een signaal met ruis en kan variëren over tijd. De vraag welke aanpak het meest geschikt is voor het aansturen van de prothese over meerdere dagen moet daarom beantwoord worden. In dit onderzoek zijn drie verschillende modellen om het knie-moment te voorspellen onderzocht. Het eerste model, *machine learning* (ML), bevatte een *convolutional neural network* (CNN) welke sEMG direct vertaalde naar het knie-moment. Het tweede model gebruikte een *neuromusculoskeletal model* (NMS) welke sEMG, spier-pees lengtes en moment armen als invoer gebruikte om het knie-moment te berekenen. Het derde model (*Hybrid*) gebruikte een CNN om sEMG te vertalen naar specifieke activaties van spier-pees eenheden, die samen met spier-pees lengtes en moment armen in NMS-componenten werden gebruikt om het knie-moment te berekenen. Metingen over meerdere dagen werden uitgevoerd op negen gezonde proefpersonen welke niet-gewichtsdragende activiteiten uitvoerden. Uit de resultaten blijkt dat ML het beste presteert in het algemeen en op elke dag (NRMSE 10.1%±5.1%). Het Hybride model (NRMSE 13.4%±5.2%) presteerde beter dan NMS (NRMSE 15.5%±7.0%). Alle modellen hadden enkel significante verschillen in prestatie tussen de eerste dag en alle andere dagen. Deze resultaten laten de meerwaarde van machine learning zien in het aansturen van een transfemorale prothese.

[1]English summary is provided as an abstract on page 6.

Contents

sEMG to Knee Torque mapping using (Hybrid) Neuromusculoskeletal Modelling and Machine Learning

Marijke Zondag BSc.

Abstract— **Proportional control using surface electromyography (sEMG) enables more intuitive control of a transfemoral prosthesis. However, sEMG is a noisy signal which can vary over time, giving rise to the question what approach is most suitable for multi-day control. In this study we investigated three different modelling frameworks to estimate knee torque. The first model, machine learning (ML), contained a convolutional neural network (CNN) which mapped sEMG to knee torque directly. The second employed a neuromusculoskeletal model (NMS) which used sEMG, muscle tendon unit lengths and moment arms as input to compute knee torque. The third model (Hybrid) used a CNN to map sEMG to specific muscle tendon unit activations, which were used together with muscle tendon unit lengths and moment arms in NMS components to compute knee torque. Multi-day measurements were conducted on nine able-bodied participants who performed non-weight bearing activities. Results show that ML had the best performance in general and on each day (NRMSE 10.1%±5.1%). The Hybrid model (NRMSE 13.4%±5.2%) was able to outperform NMS (NRMSE 15.5%±7.0%). All models had only significant performance differences between the first day and all other days. These results show the added value of machine learning in the control of a transfemoral prosthesis.**

III. INTRODUCTION

Limb loss is amongst the most physically and physiologically traumatizing events, leaving people less mobile and at risk for loss of independence [1]. In the Netherlands alone, the incidence rate of major lower limb amputations is approximately 7.7 per 100,000 person-years [2], [3]. Over the past few decades, several transfemoral prostheses have been developed that can be fitted to an amputee to regain mobility and independence during activities of daily living [4]. However, the existing solutions each have their own limitations.

The use of passive prostheses, which are not capable of generating net power, is limited by the amputee's strength and positional awareness [5], [6]. Furthermore, tasks like climbing stairs require generation of additional energy at the knee joint, which a passive prosthesis cannot do. Active (powered) prostheses can overcome some of these limitations [7]. The most common strategies in active prostheses are gait mode (activity) recognition in combination with low-level impedance control [7]. Activity recognition provides intuitive control of the active prosthesis. However, wrongly recognized activities can cause the amputee to fall, or for example not being able to climb stairs.

The EU Horizon 2020 Research and Innovation Project MyLeg aims to develop a new generation of powered transfemoral prosthetic legs [8]. Voluntary proportional (direct) control outside of pre-defined control schemes is not yet commercially available in a lower limb prosthesis to our knowledge. Direct control is desired since it provides more intuitive control for the user. Furthermore, it can be beneficial to apply direct control within a discrete state, for example to assist in sit-to-stand transitions, to reposition the prosthesis whilst in sitting-state, or to simply control the speed of a motion. Voluntary control can be realised by detection of movement onset and the mapping of the muscle characteristics to the corresponding joint torques. These joint torques will then serve as the control signal for the prosthesis. Movement onset can be detected by measuring muscle activity using electromyography (EMG), up to 138 ms in advance when the prosthesis leads [9]. Since the MyLeg project aims to employ myoelectric sensors in the prostheses, the lower limb joint dynamics can be predicted using EMG signals of the lower limb musculature.

Two methods exist that predict joint torque from sEMG and kinematics: machine learning (ML) and neuromusculoskeletal modelling (NMS), which will be explained in more detail.

A. *Machine Learning*

Multiple studies use some version of a neural network with fair performance for the prediction of knee angles in able-bodied subjects [10], [11], [12], [13], [14]. Shallow neural networks require features from data as input, which have to be determined by the researcher first. Saranya et al. for example used the root mean square values of eight EMG channels as feature input of their neural network [12]. Huang et al. proposed a deep-recurrent neural network for prediction of knee joint angles in real-time [10]. The model used sEMG signals together with IMU data from different activities and showed a mean squared error of 8.60°. Gautam et al. used a Long-term Recurrent Convolution Network to classify movements and predict their corresponding knee joint angles, based on sEMG [11]. They reported an average mean absolute error of 8.1% in the knee angle prediction of healthy subjects. Saranya et al. [12] developed a (back propagation) neural network to estimate knee range of motion, using sEMG signals as input. A mean squared error of 0.146±0.197°and 0.098±0.129°were found for knee flexion and extension respectively. Zhang et al. developed an artificial neural network for the prediction of ankle joint torque from sEMG [15]. RMSE values ranging within 0.01 and 0.1 Nm/kg were found for ankle plantar- and dorsiflexion.

All these studies indicate that machine learning is a valuable tool in predicting knee torque and/or angle.

However, current machine learning decoders might produce unrealistic estimates (outside of the physiological plausible space) in conditions they are not trained in [16]. Next to this, these methods highly rely on correct electrode placement and are thus sensitive to changes in conditions. Training data is also usually limited relative to the complexity of the models, which makes it difficult to obtain a satisfactory generalization performance [17]. The robustness to EMG electrode placement, differences in EMG signals (quality) and activity performance is thus questionable and needs to be evaluated on more data. With shallow neural networks, features have to be determined by the researcher, making it semi-subjective what features are relevant. A deep-learning method, such as a convolutional neural network (CNN), determines for itself what features of the input are relevant [18], [19]. It makes use of weight sharing, so that the number of parameters that needs training is reduced, resulting in improved generalization, robustness and less sensitivity to overfitting [18], [19]. Using a CNN instead of a shallow neural network gives rise to the possibility of a closer estimate of the relation between sEMG and joint torque.

B. Neuromusculoskeletal modelling

A neuromusculoskeletal model (NMS model) consists out of multiple components that model the underlying process of biomechanical movement. It uses sEMG and muscle characteristics from a musculoskeletal model to predict specific muscle tendon unit (MTU) activations with corresponding forces and resulting joint torques. NMS modelling was designed to gain insight in this underlying process to characterize movement function and how it alters with pathology [20].

An NMS model can provide system robustness since any joint moment estimate must always exist within the musculoskeletal model operational space and be therefore physiologically plausible [16]. Another benefit of using an NMS model is that it provides insight in the underlying process of biomechanical movement, whereas ML does not. Several studies exist that employ an NMS model to predict joint torque [16], [17], [20], [21], [22], [23], [24].

Sartori et al. [16] have succeeded in controlling a wrist-hand prosthesis by real-time neuromusculoskeletal modelling. Joint torque was predicted, using sEMG and prosthesis angles as input, and translated into low-level control of the prosthesis. They executed a virtual reaching test in which subjects always reached targets using linear trajectories, thereby successfully actuating a single DOF at a time with high precision. Path similarity was always accomplished with $R^2 > 0.98$ across all targets and subjects. Durandau et al. [21] were able to predict lower limb exoskeleton support torque, using sEMG and joint angles as input. The root mean squared error (RMSE) for the knee joint control, inside exoskeleton conditions, were 4.06±2.55°for low gain and 4.58±2.61°for high gain, with correlation coefficients of 0.90±0.16 and 0.92±0.07 respectively. Wang et al. [17] proposed a neuromusculoskeletal model with an adaptive learning method (Gaussian process regression) to learn undetermined model parameters. The RMSE value for flexion and extension of the knee joint was 1.14 Nm on the validation trial. Zhang et al. [15] used an NMS model to predict ankle joint torque with RMSEs ranging from 0.04 to 0.18 Nm/kg for ankle plantar- and dorsiflexion.

However, to our knowledge, no system that uses an NMS model exists that is commercially available. This may be because there is no compelling product yet. Technology is needed that is unified, simple to use, robust, fast and accessible [17], [25]. The NMS modelling workflow can be complex and difficult to work with. Furthermore, the mapping of sEMG to muscle tendon unit activation is based on models that are difficult to validate because activations cannot be measured directly. Another model might capture the relation between sEMG and MTU activation better. Robustness of NMS models has to be tested to a greater extent; on multiple subjects and on multiple days.

C. Hybrid modelling

Since both ML and NMS models show several shortcomings, the question arises if combining these two methods into a hybrid version will provide better joint torque predictions. Saxby et al. proposed several ideas to support neuromusculoskeletal modelling with machine learning [25]. Machine learning can decrease computational demands in physics-based modelling. It can be used for feature extraction from measures of muscle activation and to synthesize missing data.

Cimolato et al. developed a machine learning driven NMS model to predict lower limb joint torque from sEMG and IMU data [26]. The model is used for the control of a lower limb prosthesis during regular gait. They used a Gaussian Mixture Regressor to generate a complete set of sEMG signals, starting from the supposed residual subset of available sEMGs. These sEMG signals were then used as input for a calibrated NMS model to predict joint torque, with an average normalized root mean squared error (NRMSE) of $24.0\% \pm 11\%$.

Xu et al. developed an sEMG-based elbow joint torque estimation strategy using a Hill-Type Muscle Model and a neural network [27]. The neural network was used to estimate muscle activation which was used as input for the Hill-Type model. System identification from sEMG signals was used to estimate the elbow angle. The average RMSE over trials is 1.45 Nm. Only one subject was used and three trials were conducted on the same day.

These studies show promising results. However, no study has been done that evaluates the implementation of a hybrid model during non-weight bearing activities of the lower limb. Non-weight bearing means that the subject does not carry its own weight like it does in standing and walking. Furthermore, robustness of such a hybrid model must be evaluated.

D. *Contribution*

The main goal of this study was to find the most suitable model to predict knee torque from sEMG, to be used for multi-day control of a transfemoral prosthesis in non-weight bearing situations. To our knowledge, a convolutional neural network was not used before for this purpose. Furthermore, a Hybrid model as proposed in this study was not developed and implemented before or compared with both a stand-alone machine learning and neuromusculoskeletal model. Robustness of all three models to time-varying conditions such as EMG electrode placement, signal quality and performance of executing activities, was not evaluated before on multiple days. This study provides this evaluation by means of significant performance differences between days, using data from multi-day measurements. It also showed what model performed best when trained/calibrated on the first day of the measurements and tested on the remaining days. With the intended use in mind, this study gives practical knowledge about what model performs best over time in predicting knee torque in non-weight bearing situations.

IV. METHODS

Three different modelling frameworks were designed to predict knee torque using sEMG as input: a machine learning model (ML) a neuromusculoskeletal model (NMS), and a combination of ML and NMS modelling (Hybrid). This methods section is divided into 8 parts: data collection and pre-processing, the design of the ML, NMS and Hybrid models and lastly the data evaluation, statistical analysis and used software.

A. *Data collection*

Data were collected at the Roessingh Research & Development (RRD), in Enschede the Netherlands. Nine able-bodied subjects (sex: three male, six female, age: 23.8±2.4 years, length: 172.8±6.0 cm, weight 70.8±9.3 kg) were included in this dataset. Prior to the measurements, ethical approval was obtained (Medical research Ethics Committees United, Nieuwegein) and all subjects gave their informed consent. Each subject participated in four measurements: three were conducted on three subsequent days and the last measurement was four days later. The subjects were measured during the same time slot on each day. Each measurement included the same activities. The subjects were asked to perform a Maximal Voluntary Contraction (MVC) to find EMG normalisation values. These MVCs were acquired by executing several motions in which the subject exerted "maximal" force against manual resistance applied by the researcher. The exact execution is explained in Appendix I-A.

Next, the subjects were asked to perform a circuit of activities, including level-ground walking, stair ascent/descent, ramp ascent/descent, sit-stand motions and non-weight bearing (NWB) activities on a stool. The NWB activities were used in this thesis and thus further elaborated on.

The subject had to sit on a stool and lift one leg off the ground using hip flexion (knee approximately 90 degrees), as can be seen in Figure IV.1. Then, the subject had to fully extend its knee whilst keeping its foot perpendicular to its lower leg. After, the subject performed maximal plantarflexion of the ankle, followed by maximal dorsiflexion. The knee was then brought back to a knee angle of approximately 90 degrees. Then, only knee extension and flexion needed to be performed. Lastly, only ankle plantar- and dorsiflexion needed to be performed whilst keeping the knee angle at 90 degrees. After, the foot was set down on the ground and the routine was repeated with the other leg.

Fig. IV.1: Non-weight bearing setup: subject is seated on a stool with one foot slightly lifted off the ground.

The circuit in which this routine was included, was performed twenty times. Then, the routine was slightly changed for another twenty circuits: the subject had to first perform ankle plantar- and dorsiflexion, then the combination of both knee and ankle, and finish off with only knee extension and flexion.

Bipolar EMG was recorded from eight muscles on both legs: rectus femoris (RF), vastus lateralis (VL), biceps femoris (BF), semitendinosus (ST), gluteus maximus (Gmax), adductor magnus (AM), gastrocnemius medialis (GM) and tibialis anterior (TA). All EMG electrodes were placed according to SENIAM guidelines [28]. The data was acquired using the Cometa Wave electrodes at a sampling frequency of 2000 Hz. Only the RF, VL, BF and ST were included for this study since the intended application is for a transfemoral prosthesis, and these muscles extend over the knee.

Kinematics were determined using eight IMUs (Xsens Link, Enschede, The Netherlands), placed on the sternum, pelvis and bilaterally on the thigh, shank and foot of the subject. Data was recorded with a sampling frequency of 240 Hz. Joint angles were reconstructed from Xsens MVN software. Only knee joint angles in the sagittal plane were used in this study. EMG and kinematics were time synchronized and resampled to 1000Hz.

B. Data pre-processing

The data pre-processing pipeline is visualized in Appendix I-B, Figure I.1.

The smoothed rectified envelope (SRE) of the raw EMG data was obtained by high-pass filtering at 20 Hz, rectifying and low-pass filtering at 6Hz. All filters were zero-lag 2nd order butterworth filters. The SREs were normalized by values obtained from the MVCs. The SREs were windowed with a window size of 128 ms and a sampling period of 16 ms to prepare the input data for the ML model. This window has a computationally efficient size and allows for the detection of movement onset using EMG [9]. The overlap of 112 ms between windows allows for a smooth prediction. The SRE values in the window were used to generate an image of size (128,N), in which N is the number of EMG channels. This image was used as input for the ML model.

Torque reference data was obtained using OpenSim 4.1, an open source software kit to develop musculoskeletal models and make dynamic simulations [29]. First, the OpenSim lower extremity and torso model Gait2392 [30] was scaled using subject body measures to create a subject-specific model. Joint angles obtained with Xsens were analyzed with this model and used in the Inverse Dynamics tool to obtain knee joint torque. Torque offset was accounted for when the subject was seated with a foot on the ground, since no ground reaction forces were measured. Next, the torque was low-pass filtered using a 2nd order zero-lag butterworth filter with a cutoff frequency of 1Hz. The resulting knee joint torque was used to train and calibrate the developed models in this study. Muscle moment arms and muscle tendon unit lengths of the MTUs were extracted from Gait2392 as well. These parameters were used in the NMS and Hybrid models, as will be explained in section IV-D.

The reference torque and muscle analyses also had to be windowed to be used as reference and/or input in the ML and Hybrid models. The same window size and sampling period were used to match with the SREs. The average torque, MTU length and moment arm values of each window were used in the ML and Hybrid models.

C. ML model

The first data pipeline contained a machine learning model, consisting out of a CNN which maps the windowed SREs to knee joint torque. The pipeline is visualized in Appendix I-B, Figure I.2.

The CNN extracts local features from input images, using a convolutional layer with a local receptive field. It then uses layers with certain activation functions to map these features to the desired output. A Long Short Term Memory (LSTM) layer can be added to the CNN. An LSTM is an artificial recurrent neural network architecture [12], which has internal mechanisms that can regulate the flow of information and learn which data in a sequence is important to keep or discard.

For this study, no fixed CNN architecture was used. Bayesian optimization (50 runs) determined the model architecture that minimized the loss (mean squared error) between predicted and reference torque for all subjects on a training set of the first measurement day. The used hyperparameters and corresponding search space were: the number of convolutional layers {1,2,3}, the number of filters in the convolutional layer {16, 32, 64, 128}, the kernel size {3,5,7}, the number of LSTM layers {0,1}, the number of LSTM units {uniform space from 20 to 60, stepsize = 1}, the number of dense layers {1,2,3}and the number of neurons in the dense layer {uniform space from 10 to 501, stepsize = 1}.

The number of filters in a convolutional layer defines the number of windows in the convolution and the kernel size defines the size of these windows. The number of LSTM units defines the dimensionality of the output space of this LSTM layer. Next to the dense output layer, additional dense layers can be added if no LSTM layer is applied. A densely connected layer provides learning features from all the combinations of the features of the previous layer [31]. Initial investigations showed that combining LSTM and additional dense layers resulted in bad torque estimates, therefore only one of both hyperparameters was used per hyperparameter space.

Adam was used as optimizer with a learning rate of 0.01. Rectified linear units were used as activation functions for the convolutional and additional dense layers. A linear activation function was used for the final dense layer. The model weights were trained for each subject on training data of the first measurement day, making the model subject-specific. The windowed SREs were used as input for this model, the output was knee joint torque. The output was low-pass filtered with a second order zero-lag low-pass filter and a cut-off frequency of 1 Hz to obtain a smooth prediction.

D. NMS model

The second data pipeline used a neuromusculoskeletal model (NMS model) which maps sEMG to knee torque. The pipeline is visualized in Appendix I-B, Figure I.3.

The NMS model consists of three components:

1) The **neural activation component** (Figure I.3 B1) converts sEMGs into Muscle Tendon Unit (MTU)-specific activation using a second order muscle twitch model (eq. IV.1) and a non-linear transfer function (eq. IV.2) [16], [21]:

$$u_j(t) = \alpha e_j(t-d) - \beta_1 u_j(t-1) - \beta_2 u_j(t-2) \quad \text{(IV.1)}$$

$$a_j(t) = \frac{e^{A u_j(t)} - 1}{e^A - 1} \quad \text{(IV.2)}$$

in which $u_j(t)$ is the neural activation, $e_j(t)$ the SRE, α the muscle gain coefficient, β_1 and β_2 the recursive coefficients, d the electromechanical delay (15 ms), $a_j(t)$ the muscle activation, A the nonlinear shape factor and j the muscle index [32]. Parameters to be optimized in calibration for each muscle

are $\alpha, \beta_1, \beta_2, A$, which are bound by the following constraints:

$$\begin{aligned} c_1 &\in [-0.99, 0.99] \\ c_2 &\in [-0.99, 0.99] \\ A &\in [-2.99, -0.01] \\ \beta_1 &= c_1 + c_2 \\ \beta_2 &= c_1 \cdot c_2 \\ \alpha &= 1 + \beta_1 + \beta_2 \end{aligned}$$

2) The **MTU dynamics component** (Figure I.3 B3) translates the MTU activation (a_j) from the neural activation component and MTU lengths (l_{mt}) from OpenSim's muscle analysis into MTU force by employing a Hill-type muscle model.

First, muscle characteristics were extracted for each muscle using OpenSim's Gait2392 model. These characteristics are the maximal isometric force (F_{max_0}), the optimal fiber length (l_{opt_0}), the pennation angle at optimal fiber length (α_0), the tendon slack length (l_{slack}) and the maximal contraction velocity of the muscle fiber (v_m).

Two parameters of each muscle were made subject-specific. The maximal isometric force (F_{max_0}) was multiplied by the muscle strength coefficient (c_f) (eq. IV.3) and the tendon slack length (l_{slack}) was multiplied by the tendon slack coefficient (c_s) (eq. IV.4). c_s and c_f were optimized during calibration.

$$\tilde{F}_{max_0} = F_{max_0} \cdot c_f \tag{IV.3}$$

$$\tilde{l}_{slack} = l_{slack} \cdot c_s \tag{IV.4}$$

The MTU lengths (l_{mt}) from OpenSim's muscle analysis and tendon slack length were normalized by the optimal fiber length (l_{opt_0}) (eq. IV.5 & eq. IV.6).

$$\tilde{l}_{mt} = l_{mt}/l_{opt_0} \tag{IV.5}$$

$$\tilde{\tilde{l}}_{slack} = \tilde{l}_{slack}/l_{opt_0} \tag{IV.6}$$

Since these parameters were normalized, all other parameters computed using these parameters were normalized as well and are indicated with a tilde.

The tendon force ($\tilde{F}_t$) was computed using the pennation angle ($\tilde{\alpha}$) (eq. IV.7), tendon length ($\tilde{l}_t$) (eq. IV.8) and tendon strain ($\tilde{\epsilon_t}$) (eq. IV.9). $\tilde{\epsilon_t}$ was used in the pre-defined tendon force-strain relationship [33] to compute $\tilde{F}_t$.

$$\tilde{\alpha} = \arcsin(\sin(\alpha_0)/\tilde{l_m}) \tag{IV.7}$$

$$\tilde{l}_t = \tilde{l}_{mt} - \tilde{l}_m \cdot cos(\tilde{\alpha}) \tag{IV.8}$$

$$\tilde{\epsilon_t} = (\tilde{l}_t - \tilde{\tilde{l}}_{slack})/\tilde{\tilde{l}}_{slack} \tag{IV.9}$$

The fiber force ($\tilde{F}_{fiber}$) (eq. IV.13) consists out of several parts. The active fiber force ($\tilde{F_a}$) was calculated with eq. IV.11. γ is the shape factor for the parabolian active force-length relationship, which was set to 0.6 in this work [34]. $\tilde{l_m}$ is the muscle fiber length and c is the muscle activation based optimal fiber length coefficient scaling coefficient (eq. IV.10) [35]. $a_j(t)$ is the muscle activation.

$$c = 0.15 \cdot (1 - a_j) + 1 \tag{IV.10}$$

$$\tilde{F_a} = \begin{cases} 0 & \tilde{F_a} \leq 0 \\ (-1/(\gamma^2) \cdot ((\tilde{l_m}/c - 1)^2) + 1) & \tilde{F_a} > 0 \end{cases} \tag{IV.11}$$

The velocity component of the fiber force ($\tilde{F_v}$) was computed using the force-velocity relationship [33]. The fiber velocity $\tilde{v_f}$ had to be calculated using eq. IV.12, in which $\tilde{l}_{m_{old}}$ is the previous muscle fiber length, dt the timescale and v_{max} the maximal contraction velocity of the muscle fiber.

$$\tilde{v_f} = (\tilde{l_m} - \tilde{l}_{m_{old}})/(dt \cdot v_{max}) \tag{IV.12}$$

d_m, the parallel muscle damping factor set to 0.1 in this work, was multiplied with $\tilde{v_f}$ to account for muscle damping. The total fiber force $\tilde{F}_{fiber}$ was calculated using eq. IV.13. $\tilde{F}_{fiber}$ was scaled back from normalization using the maximal isometric force F_{max_0} (eq. IV.14).

$$\tilde{F}_{fiber} = (\tilde{F_a} \cdot \tilde{F_v} \cdot a_j + d_m \cdot \tilde{v_f}) \cdot cos(\tilde{\alpha}) \tag{IV.13}$$

$$F_m = \tilde{F}_{fiber} \cdot F_{max_0} \tag{IV.14}$$

The muscle fiber length ($\tilde{l_m}$) was computed by minimizing the difference between the tendon force and the fiber force, since these should be the same. This minimisation was done using sequential least squares programming, with bounds of 0 and $\tilde{l}_{mt}$.

3) The **joint mechanics component** (Figure I.3 B4) combines MTU forces F_m from the MTU dynamics component and MTU moment arms r_m from OpenSim's muscle analysis to compute the joint torque using eq. IV.15.

$$\tau = \sum_m F_m \cdot r_m \tag{IV.15}$$

The subject-specific NMS model needed to be calibrated. Bayesian optimization (400 runs) was used to adapt model parameters to minimize the error between `pyceinms` output torque and reference torque [32]. More runs were required compared to ML because this hyper optimization determines the subject-specific parameters within a large hyperparameter search-space compared to the general (not subject-specific) model architecture parameters of ML. The SREs, MTU moment arms and MTU lengths were used as input for the calibrated model. The output of the model was knee torque, which was low-pass filtered the same way as ML.

E. Hybrid model

The third data pipeline contained a hybrid model which consists out of parts of both the ML and NMS model. The pipeline is visualized in Appendix I-B, Figure I.4. A CNN was used to replace the neural activation component of the NMS model, described by eq. IV.1 and eq. IV.2. The CNN model architecture was built from the hyperparameters

found by the optimized ML model. The activation of the last dense layer of the ML model was changed from a linear to a softmax function, to limit the four output activations between zero and one. The tendon slack length coefficients and muscle strength coefficients that were optimized in the NMS model for each subject were used for the Hybrid as well. Together with the trained CNN weights, these parameters made the Hybrid model subject-specific.

Windowed SREs were used as input and mapped onto specific MTU activations by the CNN. These predicted activations were used as input for the remaining NMS model, described by eq. IV.12-IV.14. Windowed MTU lengths and moment arms were used in this remaining NMS model as well. The output of the model was knee torque, which was low-pass filtered the same way as ML and NMS.

F. *Data evaluation*

For each leg from each subject, three different models were trained. A fixed train/test split of 80/20% was made on data from the first measurement day. All models were trained and validated on the training set (80%), in which a shuffled train/validate split of 80/20% was made. All models were tested on the test set (20%) and all data from remaining measurement days. This separation in data was made for the intended application: it would be ideal to train a model on just one day and to have it perform well on every other day. With this method, robustness of all models against varying circumstances could be tested.

The performance metric used for this study was the normalized root mean squared error (NRMSE (%)), calculated by eq. IV.16. $\hat{x}$ denotes the predicted datapoint, with x as reference, N equals the total number of datapoints of which t indicates one specific datapoint. $max(x_t)$ and $min(x_t)$ are the highest respectively lowest torque reference value of all trials of one leg and are used to normalize the data.

$$NRMSE(\%) = \frac{\sqrt{\frac{1}{N}\sum_{t=0}^{N}(\hat{x}_t - x_t)^2}}{max(x_t) - min(x_t)} \cdot 100\% \quad \text{(IV.16)}$$

Robustness was indicated by not-significant differences at the 0.05 significance level, in mean NRMSE values on different measurement days.

G. *Statistical Analysis*

To analyze the performance of the models, NRMSEs of all legs, of all trials from separate days were computed with each model. A Mixed Model analysis with Šidák correction was used to determine significance ($\alpha = 0.05$) between the general performance of each model compared to one another and the performance of each model over time, compared by time and by model. A log transformation was performed on the NRMSE values to get normally distributed data.

H. *Software*

All models were built in Python 3.7.7 [36]. The machine learning components were built using Tensorflow 2.4.0 [37] and hyperparameter optimization was done using Optuna 2.2.0 [38]. The Mixed Model analysis was performed using IBM SPSS Statistics Version 26.0 [39].

V. RESULTS

In this section, the optimized CNN architecture and performance of all three models are explained. Figure V.1 reports the knee torque predictions of one subject's leg, for each model and each day separately. Mean and standard deviation (SD) NRMSE values of all subjects of all trials are visualised in Figure V.2, for each model and each day separately as well. The performance of all three models is divided into three parts, as shown in Figure V.2. The first is the performance of all models in general, in which the average NRMSE of all trials from all days per model was computed. The second is the model performance over time, in which average NRMSE values per day are shown for each model. The last is the model comparison per day, in which average NRMSE values per model on a certain day are shown for each day.

A. *ML hyperparameters*

The final optimized model architecture has the following hyperparameters: 2 convolutional layers with 16 respectively 32 filters and kernel sizes of 5, one LSTM layer with 25 LSTM units and one final dense layer with linear activation.

B. *General model comparison*

The model comparison is shown in Figure V.2a. ML has the lowest overall mean NRMSE (10.1±5.1%) compared with Hybrid (13.4±5.2%) and NMS (15.5±7.0%). Significant differences ($p<0.0001$) were found for the difference between ML and NMS (5.4%) and between ML and Hybrid (3.3%). The difference between NMS and Hybrid (2.1%) was significant as well (p=0.042).

C. *Model performance over time*

The model performance over time is shown in Figure V.2b. The overall lowest mean NRMSE was found for ML on day 1 (5.9±1.8%). ML showed the highest mean NRMSE on day 3 (11.3±4.9%). Day 2 and 7 mean NRMSE values were 9.7±5.2% respectively 10.3±5.0%. For Hybrid, the lowest mean NRMSE was found on day 1 (9.2±2.9%) and the highest on day 3 (14.9±5.2%). Day 2 and 7 had mean NRMSE values of 12.5±5.1% respectively 13.6±5.1%. NMS had the highest overall mean NRMSE, found on day 3 (16.8±8.0%). The lowest mean NRMSE value of the NMS model was found on day 1 (11.1±3.2%). Day 2 and 7 mean NRMSE values were 14.6±6.3% respectively 16.2±6.8%.

Significant differences between day 1 (fixed effect) and other days were found (p=0.006), for both Hybrid and ML ($p<0.0001$), and NMS as well (p=0.011). For ML,

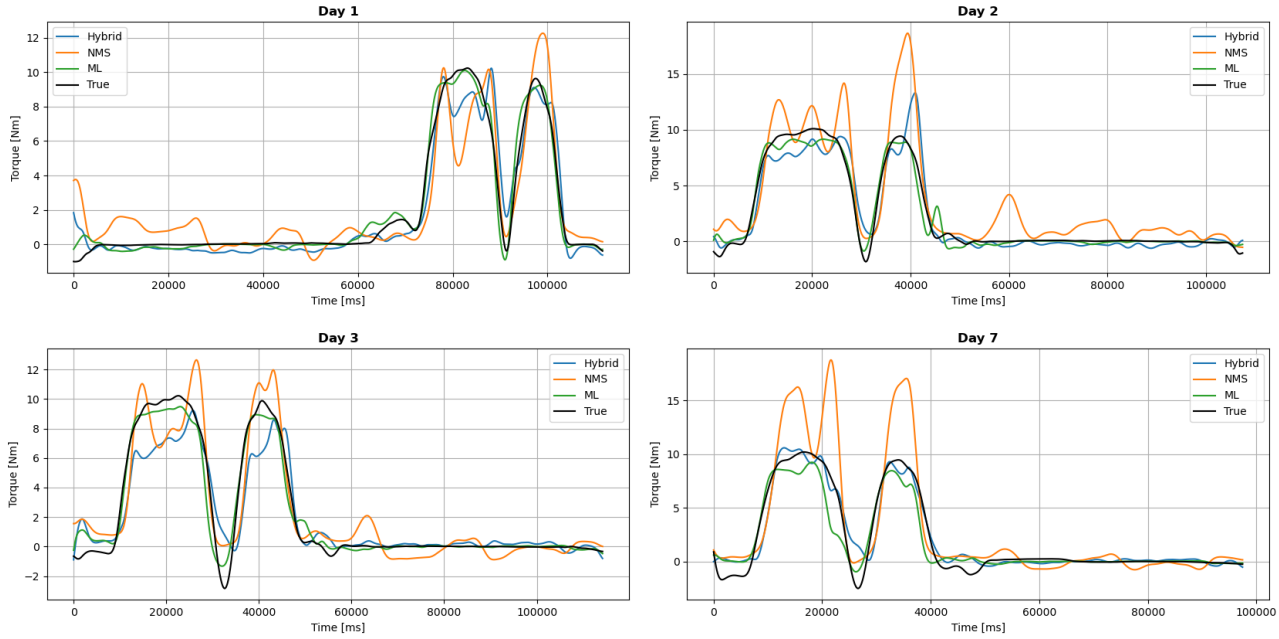

Fig. V.1: Example of model performances on one subject's leg during one trial for each day.

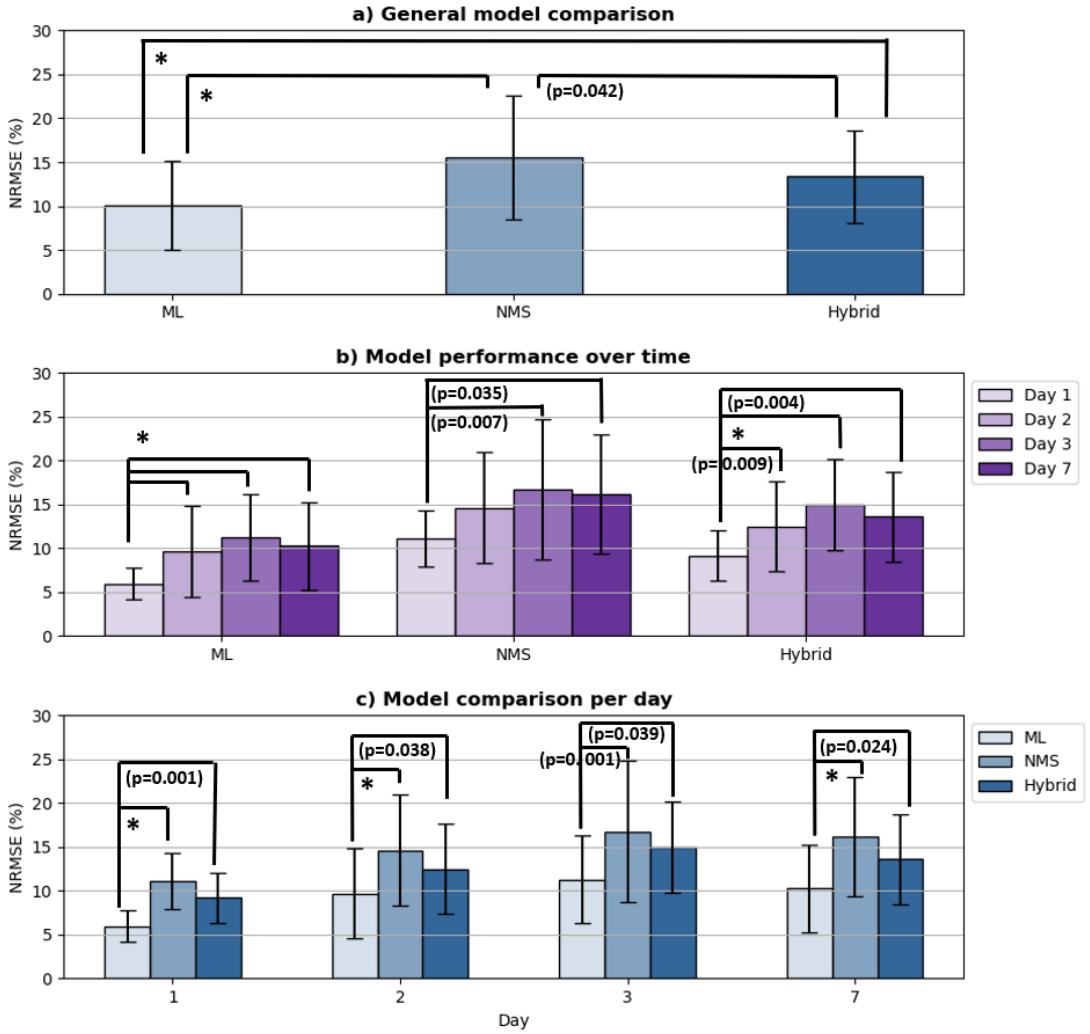

Fig. V.2: Normalized Root Mean Squared Errors (NRMSEs) mean+SD by model and day. a) shows the overall performance differences per model, b) the performance of each model investigated over days and c) the performance of each model on a day compared to other models. Significant differences between specific models or days are indicated with significance values. Significance levels of ($p<0.0001$) are indicated with an asterisk.

these differences are approximately 3.8% (day 2 - day 1), 4.4% (day 3 - day 1) and 5.4% (day 7 - day 1), with $p<0.0001$ for all. For NMS, significant differences were found between day 1 and day 3 (5.7%, p=0.007) and between day 1 and day 7 (5.1%, p=0.035). For Hybrid, significant differences were found between day 1 and 2 (3.3%, p=0.009), day 1 and 3 (5.7%, $p<0.0001$) and day 1 and 7 (4.4%, p=0.024).

D. *Model comparison per day*

The model comparison per day is shown in Figure V.2c. Significant differences between ML (fixed effect) and both NMS and Hybrid were found ($p<0.0001$). On day 1, the difference between ML and NMS was 5.2% ($p<0.0001$) and between ML and Hybrid 3.3% (p=0.001). The significant difference on day 2 between ML and Hybrid was 2.8% (p=0.038) and between ML and NMS 3.9% ($p<0.0001$). On day 3, significant differences were 5.5% between ML and NMS (p=0.001) and 3.6% between ML and Hybrid (p=0.039). Day 7 had significant differences between ML and NMS (5.9%, $p<0.0001$) and between ML and Hybrid (3.3%, p=0.024). No significant differences were found between NMS and Hybrid on all days.

VI. DISCUSSION

The main goal of this study was to find the most suitable model to predict knee torque from sEMG, to be used for multi-day control of a transfemoral prosthesis in non-weight bearing situations. Three different models were developed and validated on both legs of nine able-bodied subjects, using multi-day measurements. The ML model had the overall lowest prediction error (10.1±5.1%) and performed significantly better than NMS and Hybrid on all days. This model has the most potential of being used in a transfemoral prosthesis for direct control purposes.

To the best of our knowledge, our convolutional neural network was the first of its kind able to successfully predict knee torque from sEMG input data. We also successfully combined a convolutional neural network, and a simplified Hill-type muscle model to create a hybrid model. This hybrid model was able to outperform NMS.

Results showed that ML had the best overall performance and only significant between-days differences when compared with the first measurement day. This indicates that the model shows robustness within day 2, 3 and 7, which is not completely in line with expectations from related work. Convolutional neural networks are known to have robustness issues [40], [41], [42]. For example, if electrodes are placed differently on day 2 and MVC values differ from other days, the sEMG envelope images that are created can differ too much from day 1 for the CNN to make a good prediction. This explains the significant differences between day 1 and all other measurement days. However, our findings (Figure V.2b) show no significant differences between day 2, 3 or 7, which contradicts literature. Our model is capable of performing within a range associated with robustness. A possible explanation is that the variance in sEMG envelopes, and thus the input data for the CNN, within a subject is small enough to be handled by the CNN. Another explanation is that the CNN used in this study predicts a continuous value which is low-pass filtered to smooth any outliers, improving the NRMSE and robustness.

Furthermore, related work suggested that a black-box machine learning method could predict torque values outside of a physiologically plausible space [16]. We were able to develop a machine learning model with low-pass filter, which was able to predict torques within a surprisingly small range of the reference value.

NMS had the worst overall performance and largest standard deviation, indicating largest between-subject differences. Significant differences in performance over time were found, comparable to ML (Figure V.2b). No significant differences were found within day 2, 3 and 7, indicating robustness on these days. This is not exactly in line with expectations obtained from literature; it was expected that robustness was shown on every measurement day [16], [20]. However, related work showed the model's ability to predict torque in untrained activities, thereby proving its robustness by their definition [20]. Our definition of robustness is different since it refers to time and not the model's ability to predict untrained activities. To our knowledge, little related work investigated robustness over time. Just one study calibrated the NMS model on one day and tested the model the day after, but did not test for between-days variance [16].

It is technically possible to optimize all muscular parameters in an NMS that describe the subject, although this would lead to an exponential increase in computational cost. Therefore, only the described parameters in section IV-D were optimized. A simplified NMS, without the passive force component, was used because it resulted in much better results than with the inclusion of the passive force (as shown in Appendix I-C). Furthermore, the activities performed in this study find the physiological boundaries of the knee angle and thus of the muscle parameters, which might cause this model to perform less compared to findings in literature.

Hybrid performed on average compared to ML and NMS. Similar robustness behaviour was found compared to NMS and ML: robustness was found within day 2, 3 and 7 (Figure V.2b). This can be explained by the fact that Hybrid and ML both make use of a CNN, of which its robustness was explained before. The Hybrid model outperforms NMS (Figure V.2a and V.2c). A possible explanation is that the muscle activation computed by the CNN is more accurate than the muscle activation calculated by the activation component of the NMS model. The Hybrid model thus uses the best features of both ML and NMS: the CNN is used to find a better relation between sEMG and MTU activation, and the NMS components provide information about the underlying process of biomechanical movement.

The choice to combine NRMSEs of all legs of all subject together was made to make a general comparison between

ML, NMS and Hybrid and to account for possible faulty optimizations of individual models. This makes it possible to choose one type of model to be employed for this purpose by the transfemoral prosthesis.

It should be taken into account that OpenSim's inverse dynamics torque is a prediction on itself as well and might be biased. Nonetheless, it is used by other studies as a reference as well [15], [20], [23], [43], [44], [45]. Other methods to test performance use external measurement devices such as an exoskeleton [21], torque or angular sensor [46], [47] or a prosthesis [16], which could not be used for this study. Therefore, this study uses this prediction as reference. Since no ground reaction forces were measured, the inverse dynamics tool created an offset by assuming that the feet are lifted in the air when they are actually placed on the ground. An assumption was made that when the feet were on the ground, the torque was approximately zero since no movement was present in the knee.

Furthermore, anisotropic rescaling was used to create a subject specific model from gait 2392 in OpenSim. All rescaling methods depend strongly on modelling assumptions and cannot fully take subject-specific musculoskeletal geometry into account [48]. Scaling choices can affect the estimated moment arms and muscle tendon lengths because of differences in muscle-tendon attachment sites. However, similar methods are used by related work [20], [21], [43]. Therefore, this method was also employed in this study.

Compared to related work, our study shows similar or better results. Cimolato et al. [26] found an average NRMSE of 24.0% with their Hybrid model. This study's Hybrid performed better, with an average NRMSE of 13.4%. Xu et al.[27] found an average RMSE of 1.447 Nm with their hybrid approach. This study's hybrid model has an average RMSE of approximately 1.34, using a torque scaling factor of 10 Nm as average. This study's hybrid thus results in better estimates.

A direct comparison for ML and NMS with literature however, is difficult in terms of performance metrics and purposes. For example, Zhang et al. [15] used ML and NMS in the prediction of ankle torque with RMSEs in terms of Nm/kg. For ML, our average RMSE would be approximately 0.014 Nm/kg, for NMS 0.022 Nm/kg and Hybrid 0.019 Nm/kg, using an average torque scaling factor of 10 Nm and average weight of 70.8 kg. The ANN from Zhang et al. performed in a range of 0.01 - 0.1 Nm/kg. The findings of all models of this study are in their lower range. The NMS from Zhang et al. performed in a range of 0.04 - 0.18 Nm/kg, whereas all models presented in this study showed better performance. Zhang et al. [15] also found that the NMS model predicted ankle torques better in general than the ANN, which was not in line with the findings of this study. However, they also found that the ANN predicted ankle torques better than the NMS model when trained on a large and varied set of trials. These findings suggest that the NMS model is more robust than the ANN for activities it is not trained in, but also that the ANN can outperform the NMS model when these activities are included in the training data. The latter is in line with the findings of this study. Other related work use joint angles as reference which are not normalized, and can therefore not be compared to normalized joint torque directly.

The findings of this study are promising for the use in a transfemoral prosthesis. ML proves to be the best of three models to be used for this purpose. However, future work remains to investigate if this finding extends to an online application with amputees as well.

The input of the CNNs were windowed to obtain sEMG images, which would introduce a real-time delay of the window size (128ms) + sampling period (16 ms). Output was filtered by a second order low-pass filter with a cut-off frequency of 1Hz, which would cause an additional real-time delay of 1 sample (16 ms), resulting in a total delay of 160 ms. Therefore, with the intended use in mind, it is essential to test performance of these models in real-time, using first a real-time simulation and second an actual transfemoral prosthesis. Limitations of this study are that the influence of the maximum voluntary contractions (MVCs) on the sEMG data and model performance has not been investigated, and the models are only tested on able-bodied subject data. The MVC that was performed for this study cannot be done by a transfemoral amputee. An extension of this study should include EMGs of amputees and the use of different MVCs and how they affect the model performances.

Future work will also investigate the translation of the predicted knee torque to a corresponding (prosthesis) joint angle. This will give rise to the possibility of using joint angles as reference, which are not based on a prediction like inverse dynamics. Lastly, no model was found to be robust over all days. Future work should encompass extending the calibration of the models, and thus the training data, to multiple days. This will provide knowledge about how many re-calibrations are required to improve model performance and robustness.

VII. CONCLUSION

This study provides new insight into what modelling framework performs best in predicting knee torque from sEMG data during non-weight bearing activities. Three models (machine learning, neuromusculoskeletal and a hybrid combination of both) were designed and tested on multi-day measurements to gain knowledge about the robustness of each model to time-varying parameters. Results show that the machine learning model performed best compared to the other models (NRMSE 10.1%±5.1%) and that the hybrid model (NRMSE 13.4%±5.2%) was able to outperform the neuromusculoskeletal model (NRMSE 15.5%±7.0%). All models showed only significant differences in performance between the first day and all other days, a promising finding for the robustness of each model. These results contribute to the development of a direct control scheme for a transfemoral prosthesis by providing a clear comparison of all three modelling frameworks.

REFERENCES

[1] P. Shankar, V. S. Grewal, S. Agrawal, and S. V. Nair, "A study on quality of life among lower limb amputees at a tertiary prosthetic rehabilitation center," *Medical Journal Armed Forces India*, vol. 76, pp. 89–94, 1 2020.

[2] B. Fard, P. U. Dijkstra, R. E. Stewart, and J. H. Geertzen, "Incidence rates of dysvascular lower extremity amputation changes in Northern Netherlands: A comparison of three cohorts of 1991-1992, 2003-2004 and 2012-2013," 9 2018.

[3] P. W. Moxey, P. Gogalniceanu, R. J. Hinchliffe, I. M. Loftus, K. J. Jones, M. M. Thompson, and P. J. Holt, "Lower extremity amputations - a review of global variability in incidence," 10 2011.

[4] M. H. Spaan, A. H. Vrieling, P. van de Berg, P. U. Dijkstra, and H. G. van Keeken, "Predicting mobility outcome in lower limb amputees with motor ability tests used in early rehabilitation," *Prosthetics and Orthotics International*, vol. 41, pp. 171–177, 4 2017.

[5] Y. Dabiri, S. Najarian, M. R. Eslami, S. Zahedi, H. Farahpour, and R. Moradihaghighat, "Comparison of passive and active prosthetic knee joint kinematics during swing phase of gait," in *2010 17th Iranian Conference of Biomedical Engineering, ICBME 2010 - Proceedings*, 2010.

[6] C. S. Crowe, K. A. Impastato, A. C. Donaghy, C. Earl, J. L. Friedly, and K. A. Keys, "Prosthetic and orthotic options for lower extremity amputation and reconstruction," *Plastic and Aesthetic Research*, vol. 2019, 2 2019.

[7] M. Windrich, M. Grimmer, O. Christ, S. Rinderknecht, and P. Beckerle, "Active lower limb prosthetics: A systematic review of design issues and solutions," 12 2016.

[8] Carloni R, "MyLeg: smart and intuitive osseointegrated transfemoral prosthesis embodying advanced dynamic behaviours.," 2018.

[9] E. C. Wentink, V. G. Schut, E. C. Prinsen, J. S. Rietman, and P. H. Veltink, "Detection of the onset of gait initiation using kinematic sensors and EMG in transfemoral amputees," *Gait and Posture*, vol. 39, pp. 391–396, 1 2014.

[10] Y. Huang, Z. He, Y. Liu, R. Yang, X. Zhang, G. Cheng, J. Yi, J. P. Ferreira, and T. Liu, "Real-Time Intended Knee Joint Motion Prediction by Deep-Recurrent Neural Networks," *IEEE Sensors Journal*, vol. 19, pp. 11503–11509, 12 2019.

[11] A. Gautam, M. Panwar, D. Biswas, and A. Acharyya, "MyoNet: A Transfer-Learning-Based LRCN for Lower Limb Movement Recognition and Knee Joint Angle Prediction for Remote Monitoring of Rehabilitation Progress from sEMG," *IEEE Journal of Translational Engineering in Health and Medicine*, vol. 8, 2020.

[12] S. Saranya, S. Poonguzhali, and G. Saraswathy, "Muscle activation based estimation of Knee joint angle using Surface Electromyography Signals," in *IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER)*, (Manipal), pp. 1–6, 2019.

[13] G. Wang and et al., "A Novel Estimation Approach of sEMG-based Joint Movements via RBF Neural Network," in *Chinese Automation Congress (CAC)*, (Hangzhou), pp. 1783–1788, 2019.

[14] Q. Liu, L. Ma, Q. Ai, K. Chen, and W. Meng, "Knee joint angle prediction based on muscle synergy theory and generalized regression neural network," in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM*, vol. 2018-July, pp. 28–32, Institute of Electrical and Electronics Engineers Inc., 8 2018.

[15] L. Zhang, Z. Li, Y. Hu, C. Smith, E. M. Farewik, and R. Wang, "Ankle Joint Torque Estimation Using an EMG-Driven Neuromusculoskeletal Model and an Artificial Neural Network Model," *IEEE Transactions on Automation Science and Engineering*, 2020.

[16] M. Sartori, G. Durandau, S. Došen, and D. Farina, "Robust simultaneous myoelectric control of multiple degrees of freedom in wrist-hand prostheses by real-time neuromusculoskeletal modeling," *Journal of Neural Engineering*, vol. 15, 10 2018.

[17] W. Wang, Z. G. Hou, W. Shi, X. Liang, S. Ren, J. Wang, and L. Peng, "Neuromuscular Activation Based SEMG-Torque Hybrid Modeling and Optimization for Robot Assisted Neurorehabilitation," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11954 LNCS, pp. 591–602, Springer, 2019.

[18] S. Indolia, A. K. Goswami, S. P. Mishra, and P. Asopa, "Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach," in *Procedia Computer Science*, vol. 132, pp. 679–688, Elsevier B.V., 2018.

[19] D. Weimer, B. Scholz-Reiter, and M. Shpitalni, "Design of deep convolutional neural network architectures for automated feature extraction in industrial inspection," *CIRP Annals - Manufacturing Technology*, vol. 65, pp. 417–420, 1 2016.

[20] G. Durandau, D. Farina, and M. Sartori, "Robust Real-Time Musculoskeletal Modeling Driven by Electromyograms," *IEEE Transactions on Biomedical Engineering*, vol. 65, pp. 556–564, 3 2018.

[21] G. Durandau, D. Farina, G. Asín-Prieto, I. Dimbwadyo-Terrer, S. Lerma-Lara, J. L. Pons, J. C. Moreno, and M. Sartori, "Voluntary control of wearable robotic exoskeletons by patients with paresis via neuromechanical modeling," *Journal of NeuroEngineering and Rehabilitation*, vol. 16, 7 2019.

[22] M. Sartori, M. Reggiani, D. Farina, and D. G. Lloyd, "EMG-Driven Forward-Dynamic Estimation of Muscle Force and Joint Moment about Multiple Degrees of Freedom in the Human Lower Extremity," *PLoS ONE*, vol. 7, 12 2012.

[23] M. Sartori, D. G. Llyod, and D. Farina, "Neural data-driven musculoskeletal modeling for personalized neurorehabilitation technologies," *IEEE Transactions on Biomedical Engineering*, vol. 63, pp. 879–893, 5 2016.

[24] T. Kapelner, M. Sartori, F. Negro, and D. Farina, "Neuro-Musculoskeletal Mapping for Man-Machine Interfacing," *Scientific Reports*, vol. 10, 12 2020.

[25] D. J. Saxby, B. A. Killen, C. Pizzolato, C. P. Carty, L. E. Diamond, L. Modenese, J. Fernandez, G. Davico, M. Barzan, G. Lenton, S. B. da Luz, E. Suwarganda, D. Devaprakash, R. K. Korhonen, J. A. Alderson, T. F. Besier, R. S. Barrett, and D. G. Lloyd, "Machine learning methods to support personalized neuromusculoskeletal modelling," *Biomechanics and Modeling in Mechanobiology*, vol. 19, pp. 1169–1185, 8 2020.

[26] A. Cimolato, G. Milandri, L. S. Mattos, E. De Momi, M. Laffranchi, and L. De Michieli, *Hybrid Machine Learning-Neuromusculoskeletal Modeling for Control of Lower Limb Prosthetics*. 2020.

[27] D. Xu, Q. Wu, and Y. Zhu, "Development of a sEMG-Based Joint Torque Estimation Strategy Using Hill-Type Muscle Model and Neural Network," *Journal of Medical and Biological Engineering*, pp. 1–11, 6 2020.

[28] H. J. Hermens, B. Freriks, C. Disselhorst-Klug, and G. Rau, "Development of recommendations for SEMG sensors and sensor placement procedures," tech. rep., 2000.

[29] "SimTK: OpenSim: Project Home."

[30] "OpenSim documentation: Gait 2392 and 2354 Models," 2017.

[31] M. Shaikh, G. Anand, G. Acharya, A. Amrutkar, V. Alex, and G. Krishnamurthi, "Brain tumor segmentation using dense fully convolutional neural network," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10670 LNCS, pp. 309–319, Springer Verlag, 2018.

[32] E. Ceseracciu and M. Reggiani, "CEINMS User Guide Documentation Release 0.9.0," tech. rep., 2015.

[33] F. Zajac, "Muscle and tendon: properties, models, scaling, and application to biomechanics and motor control," *Critical Reviews in Biomedical Engineering*, vol. 17, no. 4, pp. 359–410, 1989.

[34] R. D. Woittiez, P. A. Huijing, and R. H. Rozendal, "Influence of muscle architecture on the length-force diagram of mammalian muscle," tech. rep., Pflügers Archiv European Journal of Physiology, Amsterdam, 1983.

[35] D. G. Lloyd and T. F. Besier, "An EMG-driven musculoskeletal model to estimate muscle forces and knee joint moments in vivo," *Journal of Biomechanics*, vol. 36, pp. 765–776, 6 2003.

[36] "Python Release Python 3.7.7 — Python.org."

[37] "tensorflow · PyPI."

[38] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A Next-generation Hyperparameter Optimization Framework," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, (New York, NY, USA), pp. 2623–2631, Association for Computing Machinery, 7 2019.

[39] IBM Corp, " IBM SPSS Statistics for Windows," 2019.

[40] P. Arcaini, A. Bombarda, S. Bonfanti, and A. Gargantini, "Dealing with Robustness of Convolutional Neural Networks for Image Classification," in *Proceedings - 2020 IEEE International Conference on Artificial Intelligence Testing, AITest 2020*, pp. 7–14, Institute of Electrical and Electronics Engineers Inc., 8 2020.

[41] M. UliČný, J. Lundström, and S. Byttner, "Robustness of deep convolutional neural networks for image recognition," in *Communications in Computer and Information Science*, vol. 597, pp. 16–30, Springer Verlag, 2016.

[42] S. Ghosh, R. Shet, P. Amon, A. Hutter, and A. Kaup, "Robustness of Deep Convolutional Neural Networks for Image Degradations," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2018-April, pp. 2916–2920, Institute of Electrical and Electronics Engineers Inc., 9 2018.

[43] M. Sartori, D. Farina, and D. G. Lloyd, "Hybrid neuromusculoskeletal modeling to best track joint moments using a balance between muscle excitations derived from electromyograms and optimization," *Journal of Biomechanics*, vol. 47, pp. 3613–3621, 11 2014.

[44] J. S. Higginson, J. W. Ramsay, and T. S. Buchanan, "Hybrid models of the neuromusculoskeletal system improve subject-specificity," in *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine*, vol. 226, pp. 113–119, NIH Public Access, 2 2012.

[45] G. G. Peña, L. J. Consoni, W. M. dos Santos, and A. A. Siqueira, "Feasibility of an optimal EMG-driven adaptive impedance control applied to an active knee orthosis," *Robotics and Autonomous Systems*, vol. 112, pp. 98–108, 2 2019.

[46] D. Ao, R. Song, and J. Gao, "Movement Performance of Human-Robot Cooperation Control Based on EMG-Driven Hill-Type and Proportional Models for an Ankle Power-Assist Exoskeleton Robot," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, pp. 1125–1134, 8 2017.

[47] A. Ziai and C. Menon, "Comparison of regression models for estimation of isometric wrist joint torques using surface electromyography," *Journal of NeuroEngineering and Rehabilitation*, vol. 8, no. 1, 2011.

[48] L. Scheys, A. Spaepen, P. Suetens, and I. Jonkers, "Calculated moment-arm and muscle-tendon lengths during gait differ substantially using MR based versus rescaled generic lower-limb musculoskeletal models," *Gait and Posture*, vol. 28, pp. 640–648, 11 2008.

[49] D. J. Rutherford, C. L. Hubley-Kozey, and W. D. Stanish, "Maximal voluntary isometric contraction exercises: A methodological investigation in moderate knee osteoarthritis," *Journal of Electromyography and Kinesiology*, vol. 21, pp. 154–160, 2 2011.

APPENDIX

A. *MVCs*

Eight muscles needed to have normalized sEMGs and thus had to be activated via maximum voluntary contractions (MVC). The rectus femoris (RF) and vastus lateralis (VL) are one muscle group (knee extension) and the biceps femoris (BF) and semitendinosus (ST) are another muscle group (knee flexion). Furthermore, the gluteus maximus (Gmax), adductor magnus (AM), tibialis anterior (TA) and gastrocnemius medialis (GM) are activated separately.

The MVC method was inspired by the method described in Rutherford et al. [49]. The subject was standing upright, using a wall or pole for balance and was asked to perform the following exercises, for a duration of five seconds:

- **RF-VL**: Subject flexes its hip and knee to approximately 90 degrees. The researcher places its hands on the anterior side of the lower leg, just above the ankle, and applies resistance. The subject tries to extend its knee and thus (maximally) push away the hands of the researcher, whilst keeping the upper leg in the same position.
- **BF-ST**: The same initial setup as RF-VL. The researcher places its hand on the posterior side of lower leg, just above the ankle, and applies resistance. The subject tries to flex its knee whilst keeping the upper leg in the same position.
- **AM**: Subject lifts one foot off the ground. The knee is fully extended and the researcher places its hands just above the knee, on the medial side of the leg. The subject tries to pull its leg medially to the other leg whilst the researcher exerts lateral resistance.
- **Gmax**: Subject lifts one foot off the ground and keeps its knee fully extended. The researcher places its hands just below the knee, on the anterior side of the lower leg. The subject performs hip extension against the resistance. Next, the researcher places its hands on the posterior side of the lower leg. The subject performs hip flexion against the resistance.
- **TA**: Subject flexes its hip and knee to approximately 90 degrees. The researcher places its hands on top of the toes. Subject performs dorsiflexion.
- **GM**: Subject flexes its hip and knee to approximately 90 degrees. The researcher places its hands under the toes. Subject performs plantarflexion.

B. *Pipeline visualisation*

The data pre-processing pipeline is visualised in figure I.1. The three different modelling pipelines are visualised in figure I.2,I.3 and I.4.

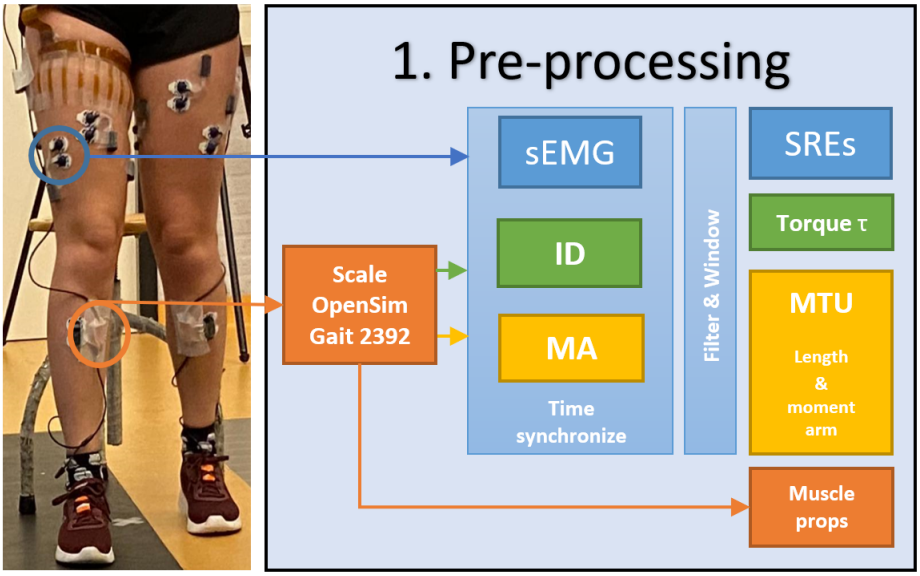

Fig. I.1: Data pre-processing pipeline. Surface EMG (sEMG, blue circle with arrows) was measured, as well as joint kinematics (orange circle with arrow). The joint kinematics and body measures were used to scale the generic OpenSim Gait2392 model to match the subjects dimensions. Using this model, the muscle properties (Muscle props) were extracted. The Inverse Dynamics (ID) tool was used to calculate the reference torque (green box). During muscle analysis (MA), muscle tendon unit (MTU) lengths and moment arms were computed (yellow box). All data was time synchronised. sEMG data was filtered to create smooth rectified envelopes (SREs). All data was also windowed to be used in the ML and Hybrid pipelines. Regular (not-windowed) data was used in the NMS pipeline.

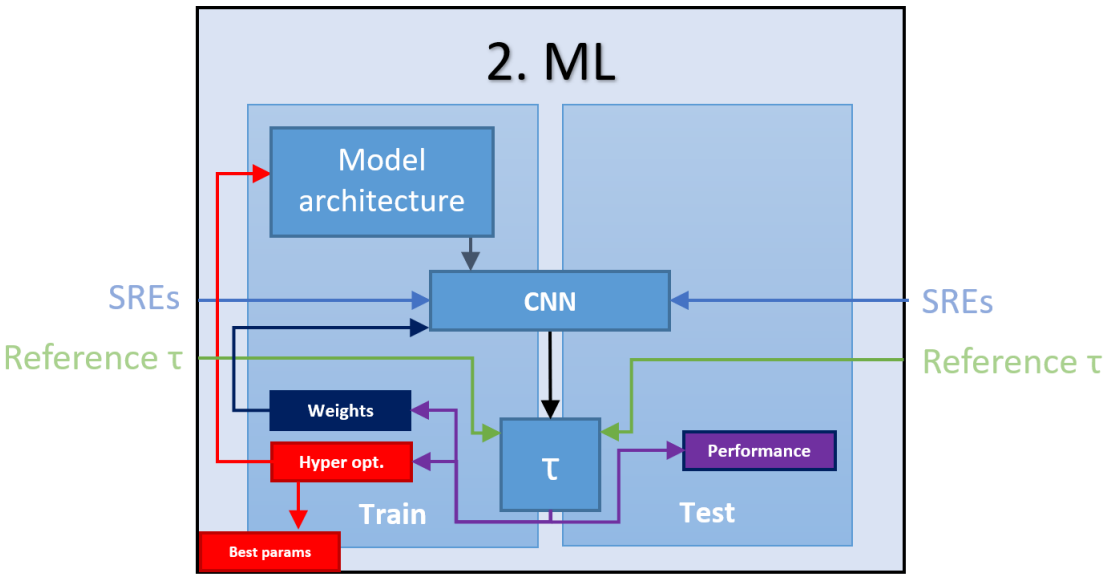

Fig. I.2: Machine learning (ML) pipeline to map sEMG to knee joint torque. The blue arrows correspond with the SREs and the green arrows with the torque from the data pre-processing pipeline. Hyper optimisation was used on data of all subjects to find the best model architecture as explained in section IV-C. The best architecture is saved and used for training and testing. Model weights are trained on training data of the subjects by minimising the loss between predicted and reference torque τ. The model is validated on part of the training data before the model is tested. The final model uses SREs as input to predict the knee joint torque, which is compared to the reference torque. The outputs of the pipeline are the performance of the model (NRMSE) and the predicted torque.

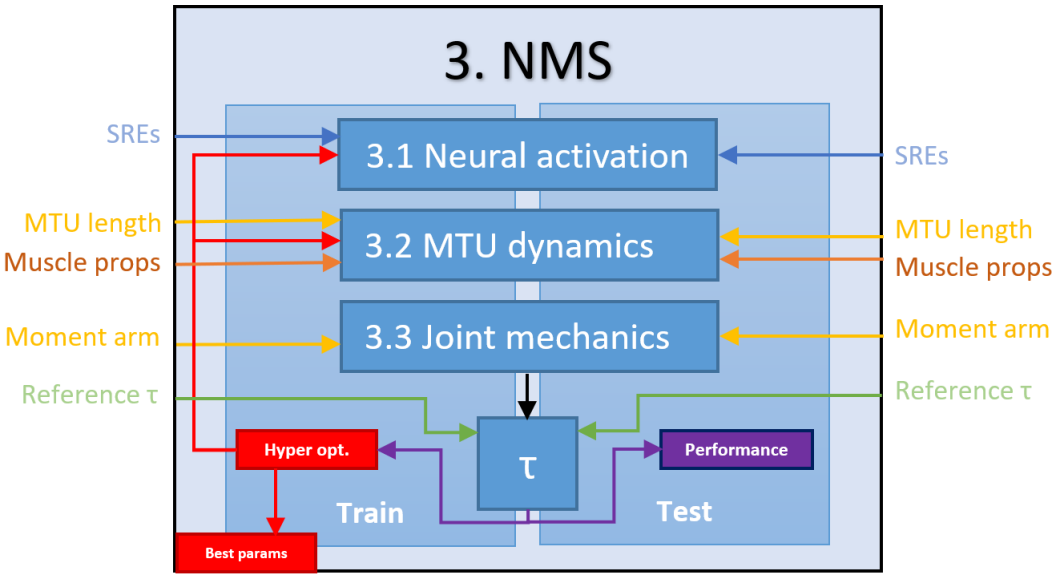

Fig. I.3: Neuromusculoskeletal (NMS) pipeline to map sEMG and muscle parameters to knee joint torque. The blue arrows correspond with the SREs, yellow arrows with the muscle tendon unit (MTU) lengths and moment arms, orange arrows with the muscle properties and the green arrows with the reference torque from the data pre-processing pipeline. During training, hyperoptimization was used to compute activation parameters, tendon slack-length and muscle strength coefficients (red arrows), by minimizing the loss between predicted and reference torque. The best parameters are saved and used to test the model. The neural activation component (3.1) takes SREs as input and computes MTU specific activations. These activations are used, together with the muscle properties and MTU lengths as input for the MTU dynamics component (3.2). This component calculates the MTU specific force, which is used as input, together with the MTU moment arms, for the joint mechanics component (3.3). The latter predicts the knee joint torque which is compared to the reference torque. The outputs of the pipeline are the performance of the model (NRMSE) and the predicted torque.

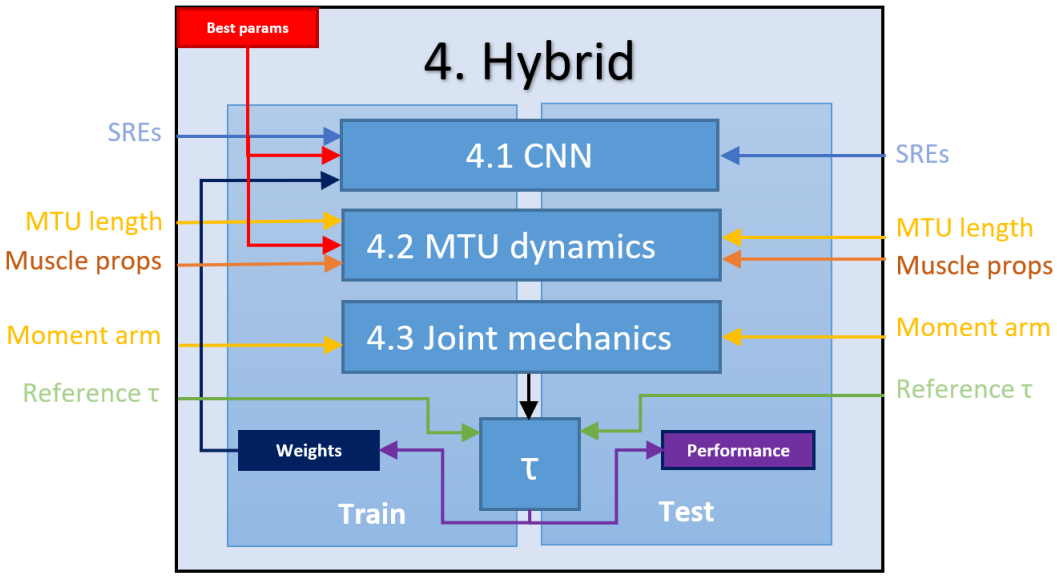

Fig. I.4: Hybrid pipeline to map sEMG and muscle parameters to knee joint torque. The blue arrows correspond with the SREs, yellow arrows with the MTU lengths and moment arms, orange arrows with the muscle properties and the green arrows with the reference torque from the data pre-processing pipeline. The optimized parameters from ML and NMS are used in this pipeline. During the training phase, the model weights of the CNN are trained by minimizing the loss between predicted and reference torque. The CNN (4.1) takes SREs as input and predicts MTU specific activations. These activations are used together with MTU lengths and muscle properties as inputs for the MTU dynamics component (4.2). This component predicts MTU force, which is used together with the MTU moment arm as input for the joint Mechanics component (4.3). The latter predicts the knee joint torque which is compared to the reference torque. The outputs of the pipeline are the performance of the model (NRMSE) and the predicted torque.

C. *CEINMS vs. pyceinms*

A comparison was made between `CEINMS` and the NMS model used in this study (`pyceinms`), both trained on 1 trial of 1 leg and tested on another trial of the same leg.

Figure I.5 shows the torque prediction from both `CEINMS` and `pyceinms` on the test trial. The result is completely off and even reversed. Therefore, more insight was required in the underlying process. For CEINMS, this was visualized in Figure I.6 and I.7. As can be seen, the activation of the extensors (rectus femoris and vastus lateralis) matches with the reference torque (Figure I.6). The generated force however, does not match with the activation, resulting in the faulty torque prediction. This particular finding was found also in `pyceinms` and indicated faulty passive force predictions as shown in Figure I.8. However, this finding could not be checked in `CEINMS` on the short-term since this was not provided as output in default settings. The fiberlength computed by `CEINMS` was according to our predictions (Figure I.7). It shortens for the extensors during knee extension, thereby generating force, and lengthens for the flexors (biceps femoris and semitendinosus) at the same time. We therefore expect that the problem is indeed the passive force for `CEINMS` as well. Since these predictions were completely off, it was decided to exclude the passive force component in the NMS model used for this study. Results of the same trial using such a simplified model are shown in Figure I.9.

Furthermore, `pyceinms` makes use of optimization methods during calibration of the model. The choice of using Bayesian optimization over simulated annealing, as used in `CEINMS`, was because it is an efficient optimizer for complex objective functions.

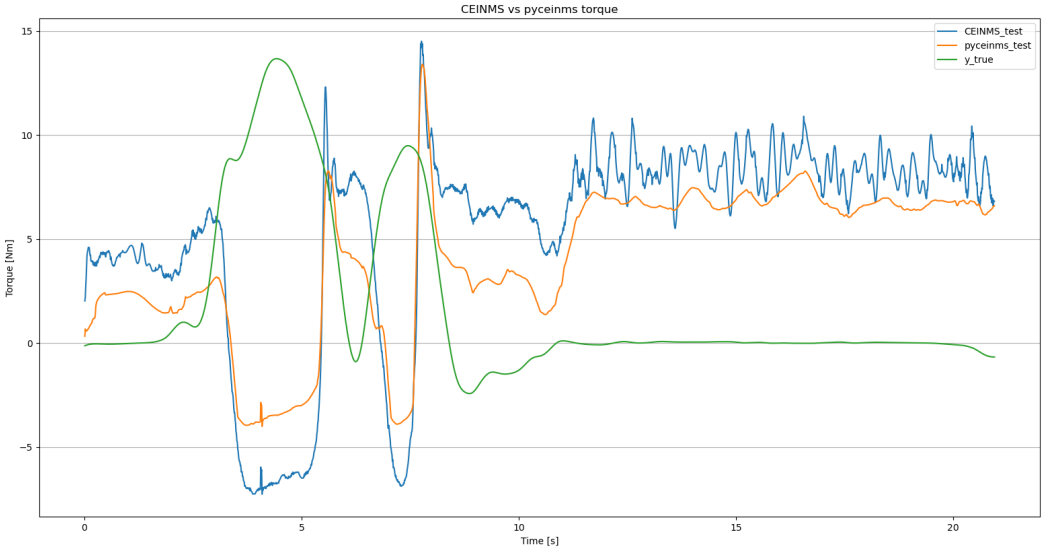

Fig. I.5: Torque prediction of test trial using CEINMS and pyceinms.

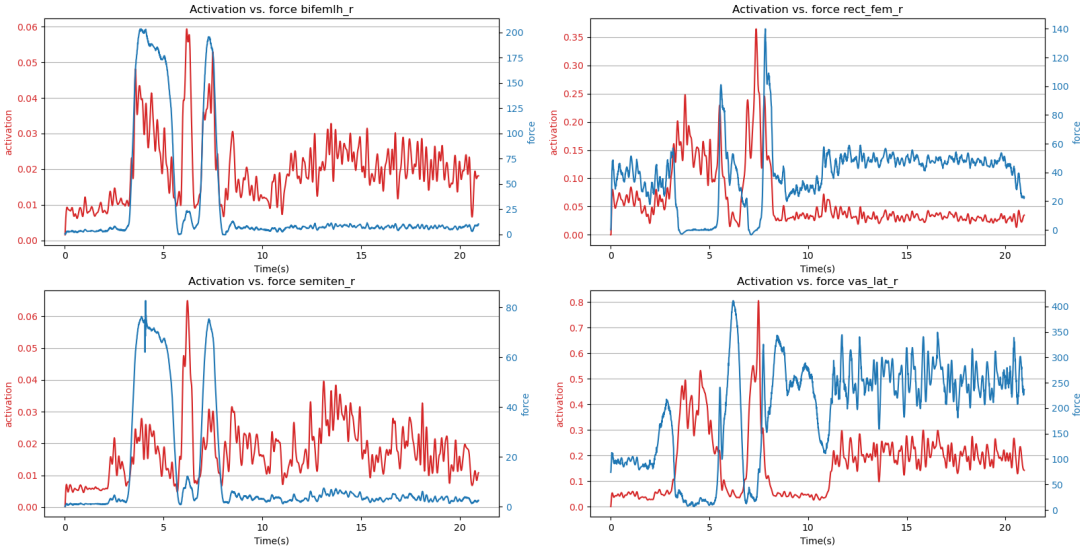

Fig. I.6: Activation vs. force, as determined by CEINMS.

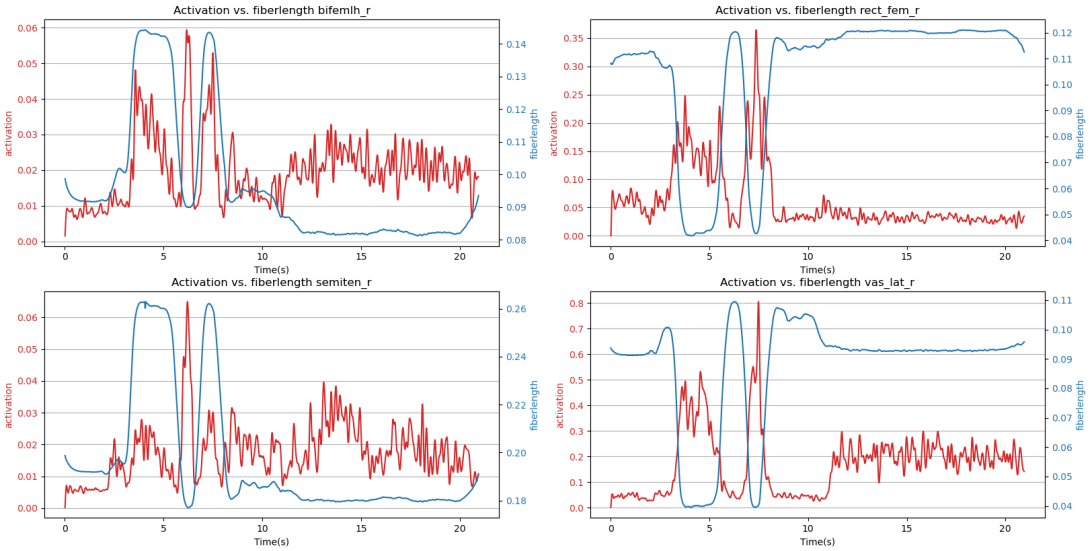

Fig. I.7: Activation vs. fiber length, as determined by CEINMS.

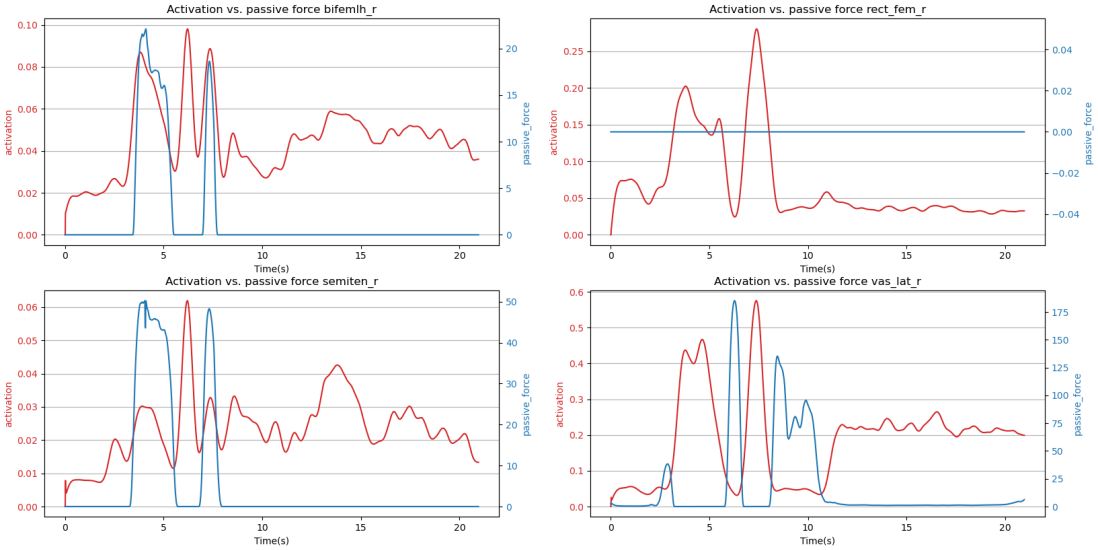

Fig. I.8: Activation vs. passive force using pyceinms

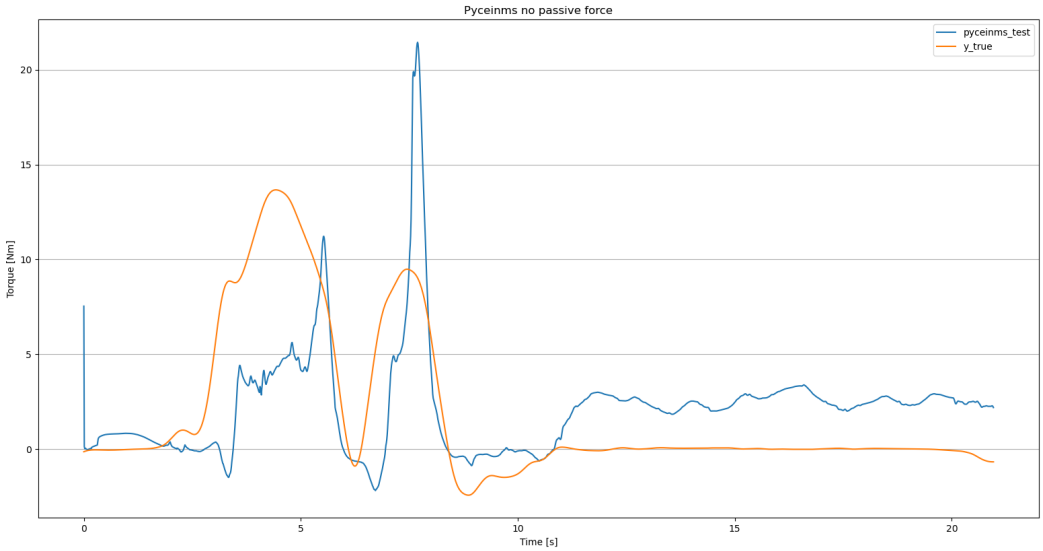

Fig. I.9: Torque prediction of test trial using pyceinms without the passive force component.

*D. **Reflection report***

In this report, I will give more insight to the work behind my article. I enjoyed the research I was able to perform; it was a tough assignment which asked for a lot of flexibility in my planning, but at the end I am proud of the result.

COVID-19
Since my thesis was carried out from July 2020 to March 2021, I had to deal with the corona measures. Working from home was luckily already established at RRD and I adapted myself quite well to this situation. However, to carry out our measurements, we had to adapt our measurement protocol. Instead of preparing a subject with two researchers, we had to prepare with just one. This took a lot more time, but it also forced me to being able to handle all anatomy and hardware by myself.

CEINMS C++
During the first months of my thesis, I investigated the possibility of using the OpenSim Toolbox CEINMS for my NMS model. I have spend approximately 2 months in figuring out the C++ source code, what XML files had to be used and how I could create these using python. I was able to run this toolbox from python. However, it was extremely slow and I had little insight in what happened in the code during execution. Robert and I therefore decided to take the functionality of CEINMS and use it for a python version: `pyceinms`. I helped Robert with my knowledge of the C++ code and together we established `pyceinms` which I used in my pipelines. I also compared CEINMS with pyceinms in the short period between my greenlight and colloquium.

OpenSim
OpenSim had to be used in my python pipelines to create subject specific models, extract muscle characteristics and parameters, and creating torque reference data. Therefore, I had to figure out the API of OpenSim and its functions in order to scale the OpenSim Gait2392 model, activate the inverse dynamics tool and muscle analysis tool, using python.

Measurements
We first did three pilots in July, of which I used data to practice and build my initial pipelines. For our actual measurements, we had to order a setup with a stair and ramp. Our schedule had to be flexible in order to fit the corona measures and the delivery of this setup. We were able to start carrying out our measurements at the end of October. However, due to hardware problems, we had to exclude some subjects and change our schedule. Therefore, the last measurements were executed at the beginning of February. This asked for some flexibility in my own planning, but with help from Robert, we formed a plan so that I could still include all subjects without creating a work load that was too high at the end of my thesis.

We performed 40 measurements in total: 10 subjects, over 4 days. One measurement (including the preparation of the subject and disconnecting the subject from the hardware) took approximately 2.5 hours. Therefore we have measured over 100 hours, without taking the set-up and clean-up into account.

Labels
All of our data (approximately 40 hours of data) had to be given labels in order to extract certain activities. I took on the responsibility of labeling all data. One subject took 7.5 hours to label. I thus spent approximately 75 hours to label all our data.

Tensors
Combining the ML and NMS model turned out to be much more difficult than I had expected. ML models use Tensors, a type of algabraic objects. The NMS model consists out of regular equations. Basically, these two models use a different language and can thus not be combined as easily as I thought. Unfortunately, there was no information available about combining such complex equations as from `pyceinms` with a machine learning model. This was both interesting, since it has not been done before, as it was frustrating since it brought me yet another problem to solve. At last, we found a solution called gradient tape, a function from Tensorflow. Training the CNN required Tensorflows `tf.GradientTape` to be used, with parts of the NMS model translated into Tensors. This way, the gradient of the loss function could be followed and applied in training the CNN.

Torque offset
My initial analysis of the results of 4 subjects indicated something went wrong with the torque reference data. OpenSim had an offset when the subject was seated on the stool with a foot on the ground since we don't measure ground reaction forces and OpenSim thus thought that the foot was "hanging" in the air. I thought I solved this by applying a mask which could identify movement based on the gradient of the torque. Unfortunately, this method did not work as well for every trial on every subject. Therefore, the reference data was faulty. We have solved this problem by creating a mask not based on torque, but on the heel marker data since this could detect when the foot was lifted from the ground. We then came across the problem that OpenSim's pelvis orientation was faulty. We have fixed this by fixating the pelvis orientation. This discovery delayed the start-up from my pipelines by approximately 1.5 weeks but it did give more insight in what to encounter when using OpenSim without measuring ground reaction forces.