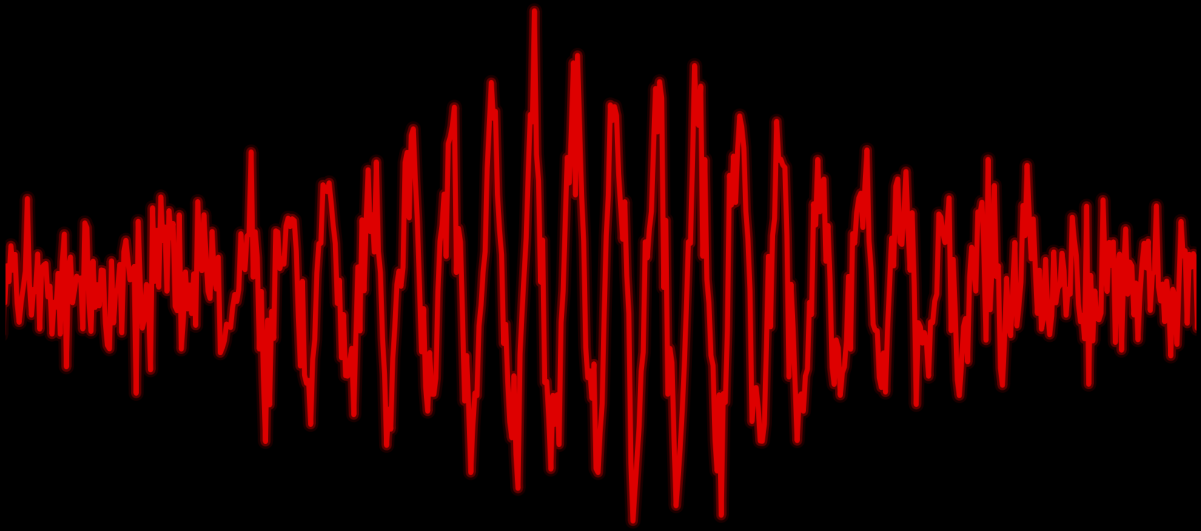
MLP-Based Torque Estimation from Decomposed HD-sEMG Signals



Harmen Reinoud Siezen

Cover illustration: Harmen Reinoud Siezen
Example of an EMG signal generated using the Python SciPy library [46]

MLP-Based Torque Estimation from Decomposed HD-sEMG Signals

Harmen Reinoud Siezen
14003945

Bachelor thesis
Credits: 18 EC

Bachelor *Kunstmatige Intelligentie*

University of Amsterdam
Faculty of Science
Science Park 900
1098 XH Amsterdam

Supervisors

Dr. A. Visser

Intelligent Robotics Lab
Informatics Institute
Faculty of Science
University of Amsterdam
Science Park 900
1098 XH Amsterdam 2

Dr.ir. A. Gogeascoechea

NEUBOTICS Lab
Department of Biomechanical Engineering
Faculty of Engineering Technology
University of Twente
De Horst 2
7522 LW Enschede

Juli 4, 2025

Abstract

It is widely accepted that the generation of force in human muscles originates from motor unit (MU) activation. High-density surface electromyography (HD-sEMG) is a non-invasive research technique capable of recording action potentials from muscles activated by MUs. As MUs are the driving source of force, and this force acts across joints to produce torque, predictions of joint torque can be made based on MU activity. However, there remains much incongruity in the relationship between MU activity and the resulting torque. This thesis investigates multiple multilayer perceptron (MLP) architectures for their ability to predict torque with the use of decomposed MU spike trains obtained with Fast-ICA, and further examines whether these models can be generalized and enhanced through transfer learning. The results show that smaller models display improved generalizability, and transfer learning reduces the need for subject-specific data and improves training time. Overall, the results demonstrate that torque can be accurately predicted based on MU activity.

Contents

Chapter 1: Introduction

In the United States, an estimated 2.3 million people are currently living with limb loss, and this number is projected to increase by 145% by 2060 [37]. Despite the growing need for medical solutions for this population, the development of effective prosthetic devices that help restore human function is hindered by our limited understanding of human-prosthesis interactions [39]. This lack of insight restricts our ability to predict how individuals adapt their movement when using prostheses.
To develop prosthetics that effectively restore function, a deeper understanding is required of how people adjust their motor behavior in response to these devices [26]. Research suggests that the effectiveness of wearable technologies such as exoskeletons and neuromodulation devices, in assisting, restoring, and enhancing motor function, depends largely on their ability to interface with the central nervous system (CNS) [33]. Humans who have experienced lower-limb loss can be aided by a prosthetic leg, which helps restore functionality and adapt to novel situations by interacting with residual neural systems (see Figure 1.1).

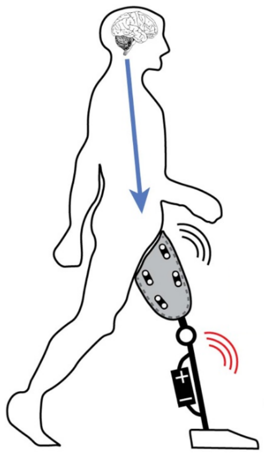


Figure 1.1: By interpreting the multichannel EMG signals transmitted through the spinal cord to the residual limb, models can estimate user intent, enabling prosthetic devices to adapt to user input and movement intent. Image obtained from [17].

The CNS acts as the primary command center of the human body, transmitting signals to the body parts via neural pathways. It consists of the brain and spinal cord, which are connected to muscles through MUs [42]. A MU links a motor neuron to a group of muscle fibers. When a MU fires, the muscle fibers are activated, leading to muscle contraction that generates force and joint torque [14].

The muscle fiber activations can be recorded through EMG, which is a technique that detects MU activity with the use of electrodes. EMG has various forms, one of which is surface electromyography (sEMG), a non-invasive method that records electrical activity from muscle fibers through the skin [44]. A limitation of sEMG is that recording through the skin can distort the original MU activity in the recorded data. Moreover, overlapping MU activity from multiple muscles aggregates in the recorded data, making it difficult to distinguish which electrical activity corresponds to each MU [11, 10, 30].

To accurately identify MUs, the number of electrodes can be increased, creating HD-sEMG, which uses tens to hundreds of electrodes to monitor the activity of multiple MUs simultaneously [42]. Using blind source separation techniques, these recordings can then be decomposed to isolate the activity belonging to individual MUs, facilitating the extraction of MU-specific signals [32]. These decomposed signals can then be used to predict intended movements [42].

Machine learning algorithms excel at detecting complex patterns in data. Because no single MU-specific signal fully explains a movement, analyzing combinations of MU-specific signals is necessary to infer intended movements. By applying machine learning algorithms to decomposed MU signals, patterns can be identified that facilitate the prediction of movements [27, 29].

Given that lower-limb amputations account for approximately 86% of all amputations globally [3], this thesis focuses on combining machine learning methods with decomposed MU activity recorded from subjects during leg-related tasks. Torque output, measured using a dynamometer, serves as a target variable for prediction based on decomposed HD-sEMG signals recorded from various leg muscles.

As data is limited in individual-specific clinical settings, there is a growing need to create generalizable models that leverage data across patients [36]. This thesis, therefore, also investigates the generalizability of these methods. Although considerable research has focused on movement prediction using decomposed MU activity from HD-sEMG, most approaches remain task and individual-specific [29, 27]. Transfer learning enables more efficient training, reducing the need for individual-specific data [34]. This enhances the applicability of neural-based prosthetics, as the models controlling these devices can adapt to new users with less data.

Chapter 2: Background

2.1 Biology

2.1.1 From brain to muscle

The human body has a hierarchical structure for controlling muscles and generating movement. At the top of the hierarchy is the primary motor cortex, located in the brain. The brain sends signals through the spinal cord, and together the brain and spinal cord form the CNS. Signals sent through the CNS are connected to the rest of the body, including the muscles, through the peripheral nervous system (PNS). Together, the CNS and PNS constitute the nervous system of the human body.

A fundamental unit of the CNS is the neuron. The neuron is in charge of generating and propagating electrical impulses called action potentials (APs) throughout the body. The type of neuron that sends signals to the muscles is called a motor neuron. There are 2 main types of motor neurons, the upper motor neuron and the lower motor neuron. The upper motor neurons are located in the brain and send signals to the lower motor neurons located in the spinal cord. The pool of lower motor neurons directly innervates the muscles, causing them to contract, which leads to the generation of force and movement. A structural representation of the nervous system in relation to controlling muscles is illustrated in Figure 2.1 [42].

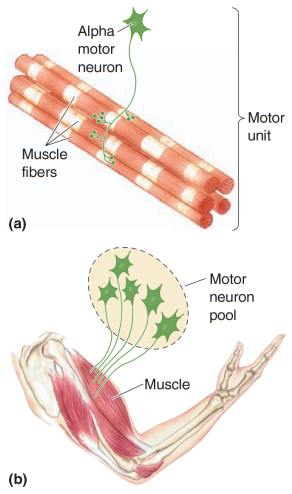


Figure 2.1: MU and motor neuron pool: (a) A MU consists of an alpha motor neuron and all the muscle fibers it innervates. (b) A motor neuron pool consists of all the alpha motor neurons that innervate one muscle. Image taken from [6].

2.1.2 Motor Units

A MU is a combination of a neural and a muscular element. The MU consists of an alpha motor neuron, its axon, and all the muscle fibers it innervates via neural connections formed by its axon branches. The MU cell is typically located in the spinal cord. The MU sends signals to the muscles through its axon. The axon branches out multiple times within the area of the muscle. When the axon enters the muscle, it branches out many more times, ultimately supplying 400 to 700 or more muscle fibers. The innervation ratio is the number of muscle fibers divided by the number of alpha motor axons supplying it [13]. MUs differ in size depending on the innervation ratio. Small motor neurons innervate fewer muscle fibers and therefore produce a lower amount of force contributing to joint torque. Consequently, large motor neurons innervate more muscle fibers and therefore produce more force and joint movement [42].

There are two categories of lower motor neurons, alpha and gamma motor neurons. Alpha motor neurons control the extrafusal fibers, which are highly contractile fibers that supply the muscle with its power. The gamma motor neurons innervate intrafusal fibers; these fibers contract only slightly. When the CNS instructs a muscle to contract, it sends signals to both the alpha and gamma motor neurons. This coordinated process is referred to as alpha-gamma co-activation [45]. Since alpha motor neurons are primarily responsible for initiating muscle contractions, we will focus exclusively on alpha motor neurons in this study.

The alpha motor neuron converts electrical signals into muscle contractions. When an alpha motor neuron receives a signal, it generates an action potential. This causes the muscle fibers to shorten, which contributes to the entirety of the muscle contracting. A group of motor neurons supplying a single muscle is called a motor neuron pool. Overall muscle contractions are generally determined by the activation of the motor neuron pool. Within each pool, small MUs are recruited first, followed by progressively to increase the force. There are many different types of MUs with great differences in size (and so innervation ratio), mechanical properties and modes of use [13].

2.1.3 Motor Unit Spike Train

Neurons transfer information via action potentials, which are brief depolarizations traveling along the axon surface. Scientists believe that the neural code is embedded in the timings of APs [42]. The firing of neurons can be represented as a series of APs called spike trains. A spike in a spike train represents a neuron firing, whereas the absence of a spike indicates the neuron is at rest [42].

A spike raster is a plot in which the firing times of individual neurons (or MUs) are visualized. The junction between the alpha motor neuron and muscle fibers can be viewed as a biological amplifier. When the alpha motor neuron fires, the muscle fibers also fire, creating a motor unit AP (MUAP). This electrical activity causes the muscle fibers to contract [5, 42]. A sequence of MUAPs is called a motor unit spike train (MUST) [42].

The signal strength of individual APs from alpha motor neurons is too weak to be detected through sEMG. However, MUAPs have a much higher signal strength, which makes them observable through sEMG [42].

The generation of force resulting from MUSTs depends on the number of active MUs and the rate at which MUAPs occur, also known as rate coding. When an alpha motor neuron fires, the muscle contracts. Multiple APs over time cause the muscle to have a sustained contraction. The amount of force a muscle can produce and the duration of its contraction vary between muscles. Under normal physiological conditions and in a non-fatigued muscle state, the MU rate coding is highly proportional to muscle force and joint torque output [42].

The amount of active MUs is also proportional to muscle force. When more torque is required, more MUs are activated to produce more force. MU activation follows Henneman's Size Principle [22]. This principle states that small MUs produce less force than large MUs. Small MUs are more easily controllable by the brain due to their small size, the geometry of their alpha motor neurons, and the smaller number of muscle fibers they innervate. This explains why tasks that require less force can be executed with higher precision [42].

2.1.4 MU recruitment and rate coding

The previous section explained that the output of torque depends on the number of active MUs and the rate at which motor neurons discharge APs [10]. The behavior of MUs is relatively fixed; when more joint torque is exerted, additional force is required, which leads to more MUs being activated (or recruited). This is commonly known as *orderly recruitment* [10]. The first MU activated stays active as long as the force does not decrease. The order of motor unit recruitment is that smaller MUs with longer contraction times are recruited first, followed by gradually larger MUs with greater force output and shorter contraction times. The exerted torque is related to the continuous activation of MUs and the discharge rate of APs from the motor neurons [10].

The sequential order of motor unit recruitment stems from the variance in MU size, as smaller MUs have a lower current requirement to reach the voltage threshold. The activation of smaller and larger MUs within a pool follows an exponential pattern, where most MUs are activated at low thresholds, while fewer are recruited at increasingly higher activation thresholds. Moreover, the additional force exerted by a muscle obtained from MU activation depends on the discharge rate of the motor neuron. The firing rate varies substantially across motor tasks. The firing rate together with the activation of more MUs are strategically used by the CNS to regulate and modulate force output and thus joint torque [10].

When muscles contract, they create force. This is caused by the muscles being connected to bones via tendons. Muscle force is closely related to movement through biomechanics. When muscles contract, they provide torque around the joint between two bones. The torque applied to a muscle is dependent on the neural drive and the stiffness of the muscle. Joint movements are dependent on a combination of neural activations. In short, synchronized neural activations lead to muscle contractions resulting in movement. If the neural drive can be extracted from EMG, movements can be predicted for prosthetic control [42].

2.2 Signal Processing

Methods to assess muscle contraction play a crucial role in predicting movement in the complex musculoskeletal system of the human body, as stated in previous sections. Muscle excitation refers to the electrical signals given to the muscles, causing them to activate, while muscle contraction involves the generation of force by the muscles activated. A deep understanding of these processes is essential to monitor and treat neuromuscular disorders, evaluate athletic performance, design rehabilitation programs and understand the workings of human movement [10].

2.2.1 Electromyography

One widely used technique to measure and analyze electrical activity produced by muscles during contraction is EMG [39, 20, 16]. With the use of electrodes, APs from contracting muscles as well as muscles at rest can be recorded.
In surface EMG, electrodes are placed on the skin overlying the muscle of interest with conductive paste. This allows for the non-invasive acquisition of recordings. The skin tissue separating the muscle and the skin acts as a low-pass filter often referred to as the volume conductor. This causes the signal to be adjusted in the recordings (See Figure 2.2) [11, 10, 30].
Intramuscular EMG is an invasive method in which fine needle electrodes are inserted into the muscle tissue for more in-depth recordings. There exist many different recording methods for surface and intramuscular EMG. These techniques can differ in the detection system, such as different electrodes, but also in the number of electrodes. Multi-electrode EMG allows for simultaneous recording over multiple locations, providing information about the anatomical location of the MU territory.

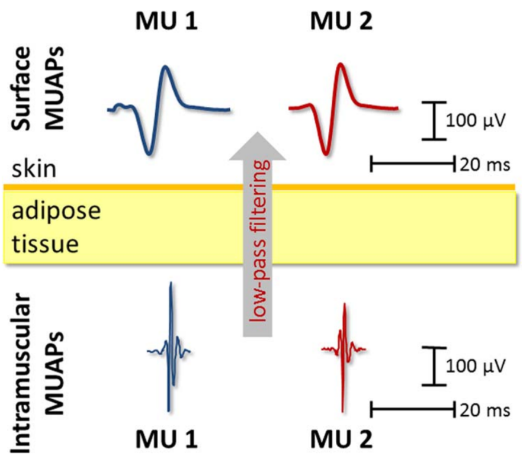


Figure 2.2: Comparison between surface EMG and intramuscular EMG. MUAPs are more distinct when recorded with intramuscular EMG, while MUAPs are less distinguishable when recorded using surface EMG [15].

For the purpose of this thesis, the following Section will give an overview of the generation of EMG signals. The sections following that will explain how these signals are recorded and processed using decomposition [10].

2.2.2 EMG generation model

Multi-channel EMG signals, whether recorded invasively or non-invasively, share the same EMG generation model and can be described as a mixture of spike trains convolved with their corresponding MUAPs (see Figure 2.3). These MUAPs have a finite duration and can therefore be modeled as a mixing process described as:

$$x_i(k) = \sum_{j=1}^{n} \sum_{l=0}^{L-1} h_{ij}(l) \cdot s_j(k-l) + \omega_i(k) \tag{2.1}$$

Here, $i = 1, ..., m$ denotes the EMG channel index, and $k = 1, ..., D$ denotes the discrete time samples, resulting in a total of D samples. The parameter L represents the duration of a MUAP in milliseconds, and n is the number of active MUs. The term h_{ij} represents the MUAP waveform (or filter) for MU j as recorded on channel i, and s_j denotes the spike train of MU j. The term $\omega_i(k)$ represents additive noise on channel i, which is assumed to be a stationary, zero-mean Gaussian process [14, 23, 15].

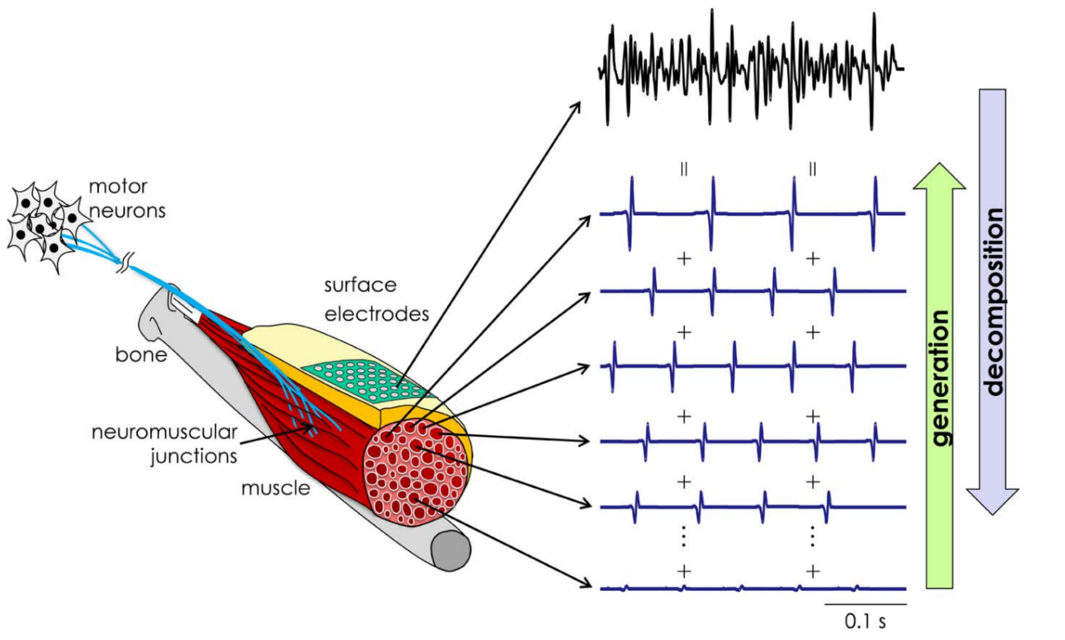


Figure 2.3: An overview of the generation and decomposition of surface EMG. Surface electrodes record the summed MU signals, which consist of MUAPS convolved with spike trains. Image obtained from [15].

The inner sum over l is the discrete convolution of s_j with h_{ij}. In general [35]:

$$x(k) = (h * s)(k) = \sum_{l=0}^{L-1} h(l) \cdot s(k-l) \tag{2.2}$$

Using this definition, the EMG generation model (see Equation 2.1) can be rewritten more compactly as:

$$x_i(k) = \sum_{j=1}^{n} (h_{ij} * s_j)(k) + \omega_i(k) \tag{2.3}$$

Since multi-channel EMG signals are typically represented as a vector across channels, the full model across all channels m can be expressed as:

$$x(k) = \begin{bmatrix} \sum_{j=1}^{n} (h_{1j} * s_j)(k) + \omega_1(k) \\ \vdots \\ \sum_{j=1}^{n} (h_{mj} * s_j)(k) + \omega_m(k) \end{bmatrix} \tag{2.4}$$

An overview of this model can be found in Figure 2.4.

The MUAP filters in matrix H act as the impulse responses that characterize how the spike trains s are transformed into the observed signals. This convolutive mixture of finite impulse response filters can be reformulated as a linear and instantaneous mixture by extending the source vectors to include the original sources and their delayed versions:

$$\underline{s}_j(k) = [s_j(k), s_j(k-1), \ldots, s_j(k-L-R+1)] \qquad j = 1\ldots, n \tag{2.5}$$

where L is the duration of an impulse response and R is the extension factor. Similarly, the observations and the noise are extended likewise [32, 14, 42, 28]:

$$\underline{x}_i(k) = [x_i(k), x_i(k-1), \ldots, x_i(k-R)] \qquad i = 1\ldots, m \tag{2.6}$$

$$\underline{\omega}_i(k) = [\omega_i(k), \omega_i(k-1), \ldots, \omega_i(k-R)] \qquad i = 1\ldots, m \tag{2.7}$$

The extension allows us to express the convolutive mixture as a linear instantaneous model. By applying the extension, the number of extracted sources increases through decomposition up to an extension factor of 16 [42]. Therefore, we can define the extended model linear and instantaneous mixture of an extended vector of sources [32, 14, 42, 28]:

$$\underline{x}(k) = \underline{H}\underline{s}(k) + \underline{\omega}(k) \tag{2.8}$$

with

$$\begin{aligned}
\underline{s}(k) &= [\underline{s}_1(k), \underline{s}_2(k), \ldots, \underline{s}_n(k)]^T \\
\underline{x}(k) &= [\underline{x}_1(k), \underline{x}_2(k), \ldots, \underline{x}_m(k)]^T \\
\underline{\omega}(k) &= [\underline{\omega}_1(k), \underline{\omega}_2(k), \ldots, \underline{\omega}_m(k)]^T
\end{aligned}$$

$$\underline{H} = \begin{bmatrix} \underline{h}_{11} & \cdots & \underline{h}_{1n} \\ \vdots & \ddots & \vdots \\ \underline{h}_{m1} & \cdots & \underline{h}_{mn} \end{bmatrix}$$

$$\underline{h}_{ij} = \begin{bmatrix} h_{ij}(0) & \cdots & h_{ij}(L-1) & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & h_{ij}(0) & \cdots & h_{ij}(L-1) \end{bmatrix}$$

where each $\underline{h}_{ij}$ is a Toeplitz-like convolutional matrix [32, 14, 42, 28].

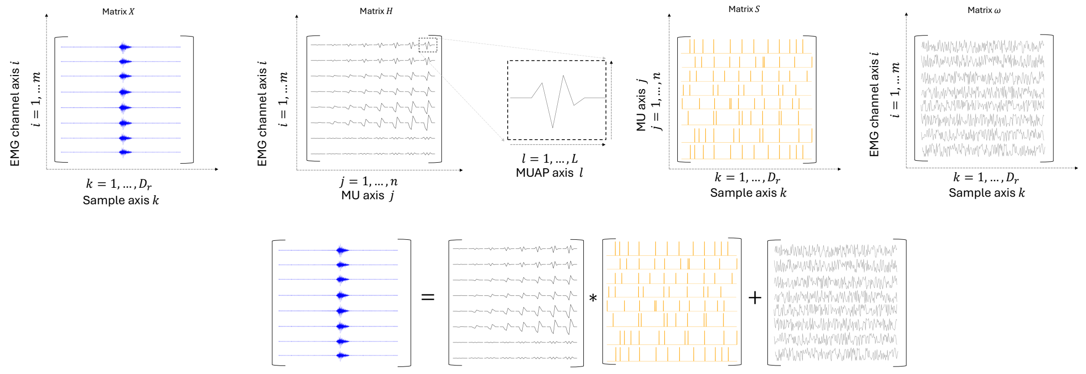


Figure 2.4: Representation of the EMG generation model: Matrix X represent the surface EMG recordings, matrix H contains the MUAP waveforms, matrix S contains the spike trains, and matrix ω contains the noise.

2.2.3 Decomposition

Because Equation 2.8 reformulates the original convolutional model (see Equation 2.2) as an extended linear instantaneous model, the spike-train matrix s can be estimated from the observations under the assumption that the sources are independent and non-Gaussian. This can be done using blind source separation, where "Blind" refers to there being no a priori knowledge about, and few assumptions regarding, the mixing matrix and the sources [42].

In order for the model to be invertible, the number of observations needs to be equal to or greater than the number of MUs, resulting in an overdetermined mixing matrix H. The observations (EMG channels) m and the sources (active MUs) n are extended, resulting in the extended model condition [42]:

$$m \cdot R \geq n \cdot (L + R) \tag{2.9}$$

HD EMG is used to greatly increase the number of observations. In practice, however, the number of MUs often exceeds to number of observations. The overdetermined condition still holds when assuming only a subset of the total active MUs are identified, and the remaining MUs contribute to the noise of the model. While the problem is not truly overdetermined, by increasing the number of observations, a greater subset of sources can be identified, resulting in a smaller noise contribution [42].

Classical decomposition methods for HD-sEMG signals rely on the uniqueness of MUAP waveforms to successfully separate sources [42]. However, the volume conduction effect of the skin (see Section 2.2.1) significantly reduces the bandwidth and distinctiveness of MUAPs, increasing the risk of waveform similarity and misclassification [15]. While increasing the number of recording channels mitigates this risk, localizing individual MUAP waveforms remains a challenging task [42].

Blind source separation techniques such as independent component analysis (ICA) enable the direct extraction of MU spike trains from the recorded signals without the need for explicit localization of individual MUAP waveforms. By extending and whitening the observed EMG signals, ICA models the signals as linear mixtures of MU source signals, simplifying the problem to estimating the unmixing matrix that recovers these sources (see Equation 2.10) [42, 32]. This approach leverages the statistical independence of MU spike trains, bypassing the need for MUAP waveform decomposition, and making it well suited for surface EMG signals affected by volume conduction and waveform overlap [42, 32].

2.2.4 Blind source separation

Independent component analysis is a BSS technique used to solve the inverse of equation 2.8. The ICA model is a generative model, meaning that the observations are linear mixture of the sources:

$$x(k) = As(k) \tag{2.10}$$

where A is an unknown $m \times n$ mixing matrix, $x(k)$ are the observations and $s(k)$ are the sources. Note that Equation 2.8 looks almost identical to Equation 2.10, where the observations x correspond to $\underline{x}$, the sources s correspond to $\underline{s}$, and the mixing matrix A corresponds to $\underline{H}$, with the noise term removed [14, 42].

The first step in the BSS pipeline is to decorrelate the extended observations by applying spatial whitening to the extended observations $\underline{\mathrm{x}}$, resulting in the extended whitened observations $z(k)$ (for details on whitening, see [28]) [32, 14, 42, 15].

ICA then aims to extract the sources from the extended whitened observations z, while blind to both the extended sources $\underline{s}$ and the mixing matrix A. This is done by solving the inverse of Equation 2.10:

$$\underline{\hat{s}}(k) = Wz(k) \tag{2.11}$$

where $\underline{\hat{s}}$ is an estimation of the extended sources, and W is the estimated unmixing matrix such that $W \approx A^{-1}$ [14, 42]. The matrix W consists of separation vectors w_i, such that $W = [w_1, ...w_n]^T$

Fast-ICA

The separation vector w_i can be obtained iteratively with numerical minimization of a predefined cost function $g(s)$. Since each source (motor unit spike train) is assumed to be independent, the separation vector w_i is used to form a linear combination of the whitened observations z, such that the resulting extended estimated source $\underline{\hat{s}}_i$ is as statistically independent as possible. The cost functions $g(s)$ are all measures of non-Gaussianity, for example when $g(s) = \log(\cosh(s))$ or $g(s) = \exp(\frac{-s^2}{2})$. The separation vector is updated by the following iteration: [15]

$$\begin{aligned} w_i(t) = \mathbb{E}[z \cdot g(w_i(t-1)^T \cdot z)] - A \cdot w_i(t-1), \\ \text{with } A = \mathbb{E}[g'(w_i(t-1)^T z)] \end{aligned} \tag{2.12}$$

where $\mathbb{E}$ denotes expectation, $g(x)$ is the first derivative of a general contrast function $G(x)$ chosen to maximize source sparsity, t is the iteration index, and z are the whitened extended observations. This is also referred to as the fixed-point algorithm, which is used to find the separation vector w_i for the i-th source [14, 15, 32].

The EMG signals acquired are decomposed into MU spike trains using the Fast-ICA algorithm (see Algorithm 8.1), which is widely used in this domain [20, 39, 16, 11, 14]. The algorithm iteratively finds the unmixing matrix W, whose rows are the separation vectors w_i, to extract the estimated extended sources $\underline{\hat{s}}$ from the extended, whitened observation z (see equation 2.11) [43].

The quantity $|w_i(t)^T w_i(t-1) - 1|$ measures the alignment between the separation vector $w_i(t)$ and $w_i(t-1)$. Let $c = w_i(t)^T w_i(t-1)$ denote the dot product between the current and previous separation vectors. Then, the quantity $|c - 1|$ quantifies the deviation from perfect alignment. The variable TOLx sets the maximum allowed deviation, ensuring that the change in the angle between $w_i(t)$ and $w_i(t-1)$ remains within acceptable bounds [14, 32].

The fixed-point algorithm calculates the separation vector w_i with the use of the cost functions $g(x)$. The cost functions are all measures of non-Gaussianity. Therefore, w_i is adjusted each iteration to become as non-Gaussian as possible. This ensures that the sources s_i are statistically independent, as the sources are assumed to be independent [14, 32].

The obtained separation vector is orthogonalized to ensure that each extracted vector points in a new direction, which prevents duplicate vectors w_i. The vectors are then normalized to unit length, so the similarity measurement depends only on the angle between the separation vectors $w_i(t)$ and $w_i(t-1)$ [14]. Once the algorithm has loops from 1 to M, a total of M candidate separation vectors are computed. However, only those candidate vectors whose extracted sources pass a similarity check (e.g., SIL > 0.9) are accepted as unique motor unit estimates. Therefore, the final unmixing matrix W consists of the $n \leq M$ accepted separation vectors $W = [w_i ... w_n]^T$

2.3 Machine learning

2.3.1 Regression

In machine learning there are three types of pattern recognition problems: unsupervised learning, reinforcement learning, and supervised learning. In unsupervised learning, input vectors $\mathbf{x}$ are available without corresponding target values. The goal of unsupervised learning is to discover groups of similar examples in the data. Reinforcement learning is concerned with finding the appropriate action in a given situation in order to maximize reward. In supervised learning, the training data consists of input vectors along with their corresponding target values [9].

Cases in which the aim is to assign input vectors to a finite set of output categories are called classification problems. If the output consists of one or more continuous variables, then the task is called regression. The simplest form of a regression task is linear regression, in which models represent linear functions of adjustable parameters [9].

In linear regression, the training dataset consists of observations $\mathbf{x}_n$ and the target values y_n, where $n = 1, ..., N$ and N denotes the total number of data points. The goal is to construct a function $f(\mathbf{x})$, typically consisting of a combination of linear functions that accurately predicts the target values y [9].

2.3.2 Neural Networks

An artificial neural network (NN) is a computational model with processing units that is able to take a set of inputs and produce a single or multiple outputs. There are numerous processing units within an NN. The architecture of the NN determines how the processing units within the network are connected. Therefore, a NN can be viewed as a processor that is composed of several micro-processors. Various types of architectures define different NNs [21].

To describe an architecture, neural networks are divided into layers. The manner in which the processing units are connected throughout the layers determines the NN. The most common type of NN in modern machine learning is the fully connected feedforward (FF) architecture, with convolutional kernels commonly used in other layers [41]. In the fully connected FF architecture, the processing units in each layer are connected to all the processing units in the following layer. The FF NN is commonly defined as data flowing through the network in one direction (see figure 2.5) [21].

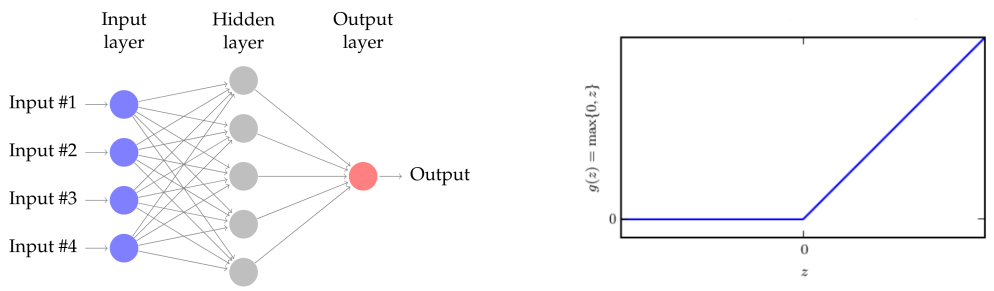


Figure 2.5: Illustration of the FF NN and the linear activation function, Images taken from [7, 21]

The connection between the FF NN layers apply weights to the input of the layer. The value resulting from applying the weight to the input is then used as input to the activation function of the next layer. A commonly used activation function for the FF NN is the linear activation function. The linear activation function makes the NN easy to optimize with gradient-based methods by but lacks nonlinearity [21].

The final layer of the NN is called the output layer. The NN trains on data, adjusting its weights. The behavior of the layers between the input and output layer is not specified. The learning algorithm must decide how to use those layers to produce a desired output. As the training data does not show the desired output for each of these layers, they are called hidden layers [21].

Multi-Layer Perceptron

A type of NN that is based on the perceptron [38], which had a major influence in the field of machine learning, is called the Multi-Layer Perceptron (MLP). As with the FF NN, the MLP can also be described according to its layers, the connections between the layers, and the activation functions. The activation function's main role is to bottleneck the output or introduce non-linearity. There are multiple choices of activation functions, but the most common ones are the sigmoid or hyperbolic tangent. The main difference between these two is the output range. The hyperbolic tangent has an output range of -1 to 1, while the sigmoid has an output range of 0 to 1, with y = 0.5 at x = 0. The MLP uses the hyperbolic tangent activation function (see Figure 2.6), whose effects become more apparent as more layers are introduced [7].

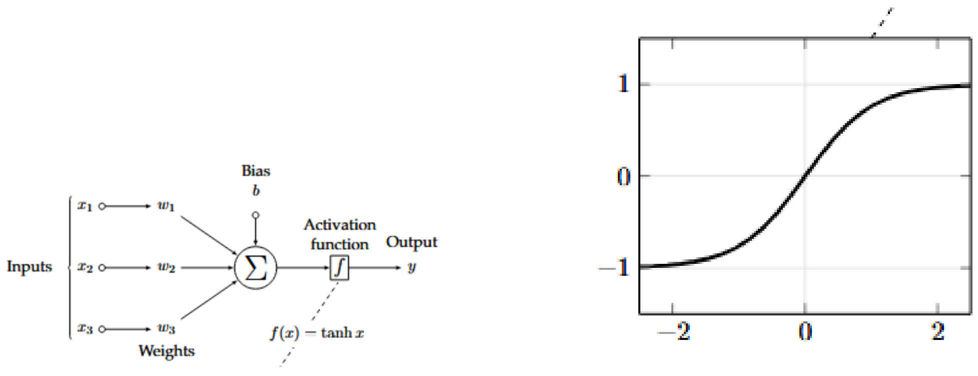


Figure 2.6: Representation of the Perceptron and the hyperbolic tangent activation function, Images taken from [7]

The usefulness of the MLP arises when multiple layers are introduced, as this enables the MLP to solve problems that are not linearly separable. The combination of the simple architecture and a powerful backpropagation algorithm makes the MLP relatively easy to train, as the number of weights to optimize remains manageable in smaller networks. The main limitation of the MLP is its simple, fully connected design. Because it does not use convolutional or other specialized layers to exploit local patterns, the number of parameters grows rapidly as more layers are added. This leads to inefficient training and a higher risk of overfitting, especially on complex, high-dimensional data [7, 21].

Despite its simplistic design, the MLP is still a widely used NN which had a major influence in the field of machine learning. It has many applications, making it the preferred model for contemporary implementations, such as decision making in convolutional NNs [7].

Generalization

Once a model is trained and able to predict the correct category or numerical value, it can be tested on data it has not seen during training to evaluate its predictive performance. The ability to correctly categorize or predict new values based on unseen examples is known as generalization, which is a fundamental concept in pattern recognition. Since the training data represents only a small portion of all possible input values a model might encounter, the model must generalize its learned patterns to novel situations when deployed in the real world [9].

Transfer learning plays a crucial role in improving model generalization. It refers to the practice of reusing the weights of one model in another to make accurate predictions. An example of this is using the weights from a pretrained DenseNet, which has been trained on millions of images. By leveraging these pretrained parameters, transfer learning reduces the computational time and mitigates data scarcity, enabling models to efficiently apply prior knowledge to a variety of problems [25, 49].

Since NNs are often trained to solve highly specific tasks, transfer learning helps extend their applicability to a broader set of problems and enables them to solve tasks more efficiently [49]. Recent developments in foundation models further address this limitation by combining multiple specialized systems. For example, multimodal systems leverage information from a large language model to enhance the prediction accuracy of a visual model, resulting in improved overall performance [1].

Chapter 3: Related Works

The control of wearable robotics is a challenging task due to the highly dynamic and unpredictable nature of human movement. Current devices often address this using sensors in a reactive manner, using sensors to detect and respond to motion as it occurs. While this approach might suffice in the repetitive context of clinical trials, it falls short for real-world environments, where systems must adapt to user intent to handle unexpected challenges. A promising solution is proactive control, which focuses on extracting movement intent before the motion takes place [28].

HD-sEMG is a technique that enables intent extraction before movement, and it is a non-invasive, relatively easy-to-acquire method for indirectly measuring overall MN activity [29, 33]. However, developing models that can generalize to new, unseen data remains a significant challenge, since the clinical training data used to train these models represent only a small fraction of real-world conditions [28].

In gesture recognition tasks, CNNs have been used to predict hand gestures by converting electrode readings into 8 by 16 pixel images derived from a 16 by 8 electrode grid. Depending on the sampling rate (2048 HZ for example), sequential images are generated and classified using majority voting over a sliding window [18]. to better respect temporal nature of EMG data, 1D CNNs have also been proposed [27]. Another variant combines spectrograms with principal component analysis (PCA) for dimensionality reduction and feeds the reduced representations into a CNN to make predictions with majority voting [50].

However, Deep models such as CNNs are prone to overfitting, particularly when trained on limited, subject-specific data [21]. Their output space is typically restricted to discrete classes, since most of these models rely on classification, which limits adaptability to novel situations.

To make predictions beyond discrete classes, blind source separation (BSS) can decompose high-density electromyography (HD-EMG) signals into individual MU signals. The decomposed MU activity provides valuable insights into muscle function mechanisms and contributes to mobility restoration and rehabilitation strategies for individuals with motor impairments [14]. Using BSS, dimensionality reduction can effectively be performed with linear discriminant analysis (LDA), which has also been combined with a support vector machine, reaching up to 99%

accuracy on a 4-class problem [2]. LDA is recognized for improving classification performance by incorporating class information [51]. Although this approach effectively boosts class-based classification, it limits generalizability, as the output is restricted to the predefined classes, overlooking movements that do not belong to any specified category [16, 20].

To address the limitations of a class-based output space, this thesis approaches the problem as a regression task. This allows the model to predict continuous values instead of discrete labels, enabling generalization beyond a fixed set of predefined classes. In addition to tackling the limitations of class-based classification, this thesis also addresses the challenge of subject-specific data by using an MLP to enhance generalization to novel scenarios for the following reasons:

- Prior work has shown that an MLP can achieve high accuracy on multiclass problems. For example, it has reached up to 82% accuracy on a 12-class problem [18], demonstrating its ability to learn complex decision boundaries, which is promising for regression tasks that require predicting continuous values.

- Deep neural networks such as CNNs are prone to overfitting on the small subject-specific datasets [21]. In contrast, the simpler architecture of the MLP allows for accurate prediction with a relatively small number of parameters. This improves training efficiency and reduces the risk of overfitting, enhancing the model's ability to generalize to unseen scenarios.

- MLPs output continuous values, unlike LDA and SVM, which depend on discrete class labels [2]. This enables adaptation to scenarios that are not class-specific, expanding the model's predictive range beyond predefined categories.

- The simple architecture of the MLP is more interpretable due to its straightforward design. Therefore, its structure can be easily modified to assess the influence of architectural changes on generalizability.

Chapter 4: Method

This thesis proposes a framework that comprises three components. The first component uses a dataset of torque measurements from three healthy subjects, performing dorsiflexion (lifting a pedal upward with the foot) and plantarflexion (pressing a pedal downward) movements. The dynamometer measurements are low-pass filtered at 2 Hz using a zero-phase, second-order Butterworth filter. To remove the offset caused by the weight of the subject's leg, the amplitude offset is calculated and subtracted from the filtered measurements [39].

The second component consists of preprocessing the HD-sEMG recordings. The HD-sEMG signals, recorded from multiple electrode grids placed over various muscles, are first high-pass filtered at 30 Hz using a zero-phase, second-order Butterworth filter. These muscle-specific signals are then decomposed into individual MUs. Each muscle yields multiple MU spike trains, which are summed to create muscle-specific feature functions [39].

The third component uses the muscle-specific feature functions as input features, with the torque measurements serving as labels. A Multi-layer perceptron is then trained to predict torque from these features (see Figure 4.1). To assess the model's robustness and generalizability, various training/test sets across subjects and tasks, as well as various MLP architectures, are evaluated. time series cross-validation is employed to ensure reliable performance [14].

4.1 Data Collection

Experiments were conducted in accordance with the Declaration of Helsinki [39]. The University Medical Center Göttingen Ethical Committee approved all experimental procedures (Ethikkommission der Universitätsmedizin Göttingen, approval number 01/10/12). Four healthy men (age: 30 ± 1.9 years, weight: 68.3 ± 1.3 kg; height: 184 ± 2.1 cm) volunteered for this investigation after providing signed informed consent [47, 39]. For this thesis, data from only three of these subjects were used. Data from one subject were excluded due to a task mismatch.

Ankle angular moments and positions were measured using a dynamometer (M3, Biodex, Medical Systems Inc., Shirley, NY, USA). These recordings were synchronized and sampled at 2048 Hz with a 256-channel EMG acquisition system (EMG-USB2, OT Bioelettronica, Torino, Italy). Two 32-channel grids (a grid

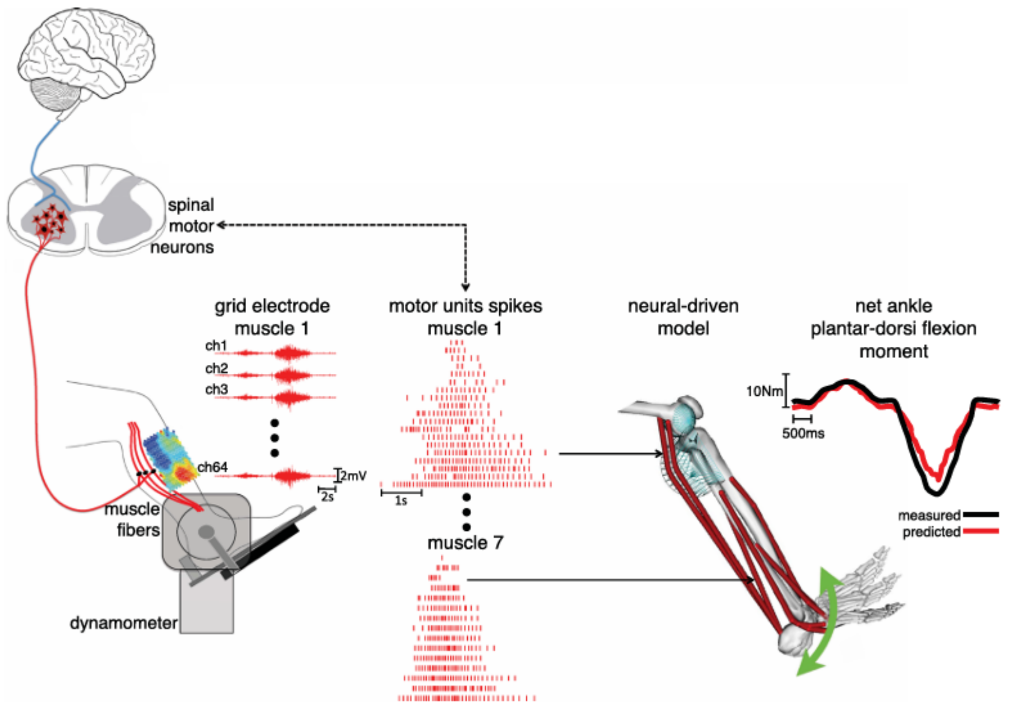


Figure 4.1: By recording moments with a dynamometer and collecting high-density EMG data, a model can be trained to predict torque from spike trains obtained through EMG decomposition. Image obtained from [39].

consisting of 32 electrodes) and three 64-channel grids (with 10-mm inter-sensor distance) were used. The HD-sEMG data were measured from the right lower leg muscles. The 64-channel grids were placed on the tibialis anterior (TA), soleus (SOL), and gastrocnemius medialis (GM) muscles. The 32-channel grids were placed on the gastrocnemius lateralis (GL) and the peroneus group, which is split up into peroneus longus (PL) (during plantar flexion) and peroneus tertius (PT) (during dorsiflexion) (see Figure 4.2). Prior to the electrode placement, the skin was shaved and lightly abraded. The grids were applied on the skin surface using conductive paste and 1-mm-thick double adhesive foam with holes corresponding to each sensing site [20, 39].

4.2 Experimental Protocol

The subjects were asked to perform a series of isometric plantarflexion and dorsiflexion contractions, which entailed moving the ankle from rest to maximum voluntary contraction (MVC) in either dorsiflexion or plantarflexion (see Figure 4.1). The protocol included four target levels at predefined percentages (30%, 50%,

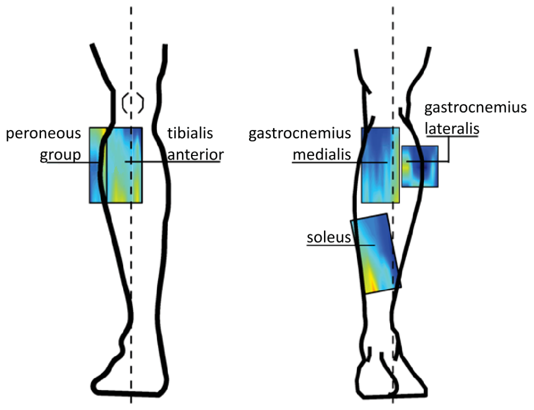


Figure 4.2: A visualization of where the high-density EMG grid electrodes were placed. The soleus and gastrocnemius muscle recording sites are located on the back of the right lower leg. The tibialis anterior and peroneus muscle recording sites are located on the front of the right lower leg. Image obtained from [39].

70%, and 90%) of MVC.

For each level, the task consisted of four repetitions. Each repetition comprised four sequential phases, including a 2-second phase transitioning from resting to dorsiflexion MVC, a 2-second phase transitioning from dorsiflexion MVC to resting condition, and two equivalent phases of plantarflexion MVC. The tasks were designed such that the slopes of the ramps varied across MVC targets; the higher the moment target, the steeper the slope [20, 39].

4.3 Data Prepossessing

The data used in this thesis consist of HD-sEMG recordings, denoted by $x_{p,q,o,t,i}(k)$, collected from three subjects ($p \in \{1, 2, 3\}$), each performing either dorsiflexion or plantarflexion movement type ($o \in \{\text{Dorsi}, \text{Plantar}\}$) at different contraction levels (30%, 50%, 70%, and 90% of MVC ($t \in \{30\%, 50\%, 70\%, 90\%\}$). During each task, HD-sEMG signals were recorded from six muscles ($q \in \{\text{TA, SOL, GM, GL, PL, PT}\}$), each equipped with a muscle-specific electrode grid of varying size.

This leads to the following EMG generation model (see Section 2.2.2 for details):

$$x_{p,q,o,t,i}(k) = \sum_{l=0}^{L_{p,q,o,t}} \sum_{j=1}^{n_{p,q,o,t}} h_{p,q,o,t,i,j}(l) s_{p,q,o,t,j}(k-l) + \omega_{p,q,o,t,i}(k) \tag{4.1}$$

where p is the subject index, q is the muscle, o is the movement type (dorsiflexion/plantarflexion), t is the task (MVC level), $i \in \{1, \ldots, m_q\}$ indexes the EMG channels for the electrode grid placed on muscle q, $L_{p,q,o,t}$ is the duration of the MUAP waveform, $n_{p,q,o,t}$ is the number of MUs, $h_{p,q,o,t,i,j}$ is the MUAP waveform from MU j at channel i, $s_{p,q,o,t,j}$ is the spike train of MU j, $\omega_{p,q,o,t,i}(k)$ is the additive noise and $k \in \{1, \ldots, D_{p,o,t}\}$ represents the sample index.

During each task, the subject was instructed to push down or pull up a pedal using the right leg. The torque data obtained from the dynamometer is expressed as:

$$y_{p,o,t}(k) = \text{torque}_{p,o,t}(k) \tag{4.2}$$

where $y_{p,o,t}(k)$ is the torque measurement at time sample k for subject p, performing movement type o at MVC level t.

As a prepossessing step, the raw HD-sEMG signals $x_{p,q,o,t}(k)$, are high-pass filtered to yield the filtered signals $\tilde{x}_{p,q,o,t}(k)$: [20].

$$\tilde{x}_{p,q,o,t,i}(k) = \text{HPF}_{30\text{Hz}}(x_{p,q,o,t,i}(k)) \tag{4.3}$$

where $\text{HPF}_{30\text{Hz}}(.)$ denotes a high-pass filter with 30 Hz cut-off frequency. This is implemented using a second-order Butterworth filter combined with the *filtfilt* function from the SciPy library in Python [46]. This filtering is applied to eliminate movement artifacts [39].

The recorded moments are initially expressed in millivolts (mV; 1 mV $= 10^{-3}$V). To convert these to torque in Newton-meters (Nm), the device manual is consulted, which specifies a scale factor of 9.76 mV per foot-pound (ft-lb) [8]. Since 1 ft-lb $=$ 1.355 Nm, this gives

$$\frac{1.355\text{Nm}}{9.76\text{ mV}} = 0.139 \text{ Nm per mV}$$

$$y^{\text{Nm}}_{p,o,t}(k) = 0.139 \cdot y_{p,o,t}(k) \tag{4.4}$$

where $y_{p,o,t}(k)$ is the raw dynamometer moment in mV, and $y^{\text{Nm}}_{p,o,t}(k)$ is the converted moment in Nm.

The signal is then low-pass filtered:

$$\tilde{y}^{\text{Nm}}_{p,o,t}(k) = \text{LPF}_{2\text{Hz}}(y^{\text{Nm}}_{p,o,t}(k)) \tag{4.5}$$

where $\text{LPF}_{2\text{Hz}}(.)$ denotes a low-pass filter with 2 Hz cut-off frequency, Moments are then averaged over a 5-s time window during which the subjects exert no torque. This average is used to remove the torque offset from the recorded moments caused by the weight of the subject's leg [39].

$$\tilde{y}^{\text{offset}}_{p,o,t}(k) = \tilde{y}_{p,o,t}(k) - \text{offset}(\tilde{y}^{\text{Nm}}_{p,o,t}(k)) \tag{4.6}$$

where offset(.) refers to the average signal over a 5-second window during which the subjects exert no torque.

As stated in Section 2.2.3, this thesis focuses on estimating source spike trains without explicitly estimating individual MUAPs. This simplification is possible because the model is transformed into a linear instantaneous form, enabling the Fast-ICA algorithm to directly extract the estimated source spike trains from the observations by estimating the unmixing matrix.

4.4 Decomposition

The Fast-ICA algorithm (see Section 2.2.4) is used to estimate the MU sources $\hat{s}_{p,q,o,t}(k)$. For this to be possible, the observations are extended to transform the original convolutive model (see Equation 4.1) into a linear instantaneous model. For this thesis, an extension factor of 16 is used, as the number of extracted sources has been reported to increase with the extension factor up to 16 [42].

$$\underline{\tilde{x}}_{p,q,o,t,i}(k) = [\tilde{x}_{p,q,o,t,i}(k), \tilde{x}_{p,q,o,t,i}(k-1), \ldots, \tilde{x}_{p,q,o,t,i}(k-R)] \tag{4.7}$$

$$\underline{\tilde{x}}_{p,q,o,t}(k) = [\underline{\tilde{x}}_{p,q,o,t,1}(k), \underline{\tilde{x}}_{p,q,o,t,2}(k), \ldots, \underline{\tilde{x}}_{p,q,o,t,m_q}(k)] \tag{4.8}$$

where R is the extension factor. This results in the following system:

$$\underline{\tilde{x}}_{p,q,o,t}(k) = A_{p,q,o,t}\underline{s}_{p,q,o,t}(k) \tag{4.9}$$

where A is an unknown mixing matrix and $\underline{s}_{p,q,o,t}(k)$ are the extended source vectors.

By whitening the extended observations $\underline{\tilde{x}}_{p,q,o,t}(k)$, we obtain $\underline{\tilde{x}}^{\text{whitened}}_{p,q,o,t}(k)$, which is subsequently used to estimate the unmixing matrix $W_{p,q,o,t}$ using the Fast-ICA algorithm $W_{p,q,o,t}$ such that $W_{p,q,o,t} \approx A^{-1}_{p,q,o,t}$

$$\underline{\hat{s}}_{p,q,o,t}(k) = W_{p,q,o,t}\underline{\tilde{x}}^{\text{whitened}}_{p,q,o,t}(k) \tag{4.10}$$

where $\underline{\hat{s}}_{p,q,o,t}(k)$ are the estimated extended sources and has dimensions $n_{p,q,o,t} \times 1$.

To obtain the estimated sources, the duplicates in the estimated extended sources are removed using k-means clustering and peak detection (for a detailed explanation of this process, see [28]), resulting in the estimated sources $\hat{s}_{p,q,o,t}(k)$ [20].

4.5 Prediction

After decomposition, the estimated sources $\hat{s}_{p,q,o,t}$ are used to predict torque using an MLP. However, since the number of estimated sources can vary per muscle, the sources are aggregated per muscle:

$$\hat{s}_{p,q,o,t}(k) = \sum_{j=1}^{n_{p,q,o,t}} \hat{s}_{p,q,o,t,j}(k) \tag{4.11}$$

Here, $n_{p,q,o,t}$ denotes the number of estimated sources for muscle q in trial t of subject p performing movement type o. This summation produces a single source signal per muscle, $\hat{s}_{p,q,o,t}(k)$, resulting in a feature vector of size 6 corresponding to the six recorded muscles $\hat{s}_{p,o,t}(k)$, allowing for consistent input to the MLP model.

Next, the summed sources per muscle are transformed into muscle-specific feature functions by applying a twitch contraction model, which predicts torque based on spike trains:

$$\begin{aligned} f_{p,q,o,t}(k) = &\left(2 \cdot \exp^{\frac{-T}{t_{\text{peak}}}}\right) \cdot f_{p,q,o,t}(k-1) - \left(\exp^{\frac{(-2 \cdot T)}{t_{\text{peak}}}}\right) \cdot f_{p,q,o,t}(k-2) \\ &+ \frac{(A_{\text{peak}} \cdot T^2)}{t_{\text{peak}}} \cdot \exp^{1-\frac{T}{t_{\text{peak}}}} \cdot \hat{s}_{p,q,o,t}(k-1) \end{aligned} \tag{4.12}$$

where A_{peak} is the peak, and t_{peak} is the time to peak (or contraction time) of the twitch. $f_{p,q,o,t}(k)$ models the torque generated by the summed spike trains $\hat{s}_{p,q,o,t}(k)$ for subject p performing movement type o at MVC level t [12, 20]. This results in a muscle-specific feature function. For this thesis, values 0.1 were used for both A_{peak} and t_{peak}, aligning with previous research on twitch functions [20].

In the recorded data, four % MVC tasks are performed. These tasks are performed in either dorsiflexion or plantarflexion, resulting in eight recordings for each subject. With the use of an MLP, the muscle-specific feature functions are used to predict the offset-corrected torque values $\tilde{y}^{\text{offset}}_{p,q,o,t}(k)$:

$$\tilde{y}_{p,o,t}^{\text{offset}}(k) \approx \text{MLP}(f_{p,o,t}(k)), \qquad \text{where } f_{p,o,t}(k) = \begin{bmatrix} f_{p,\text{TA},o,t}(k) \\ f_{p,\text{SOL},o,t}(k) \\ f_{p,\text{GM},o,t}(k) \\ f_{p,\text{GL},o,t}(k) \\ f_{p,\text{PL},o,t}(k) \\ f_{p,\text{PT},o,t}(k) \end{bmatrix} \tag{4.13}$$

Different MLP architectures are tested with 3, 5, and 9 layers, each containing 8, 16, or 32 nodes, as these configurations have been shown to yield the highest accuracy in previous studies [53, 4]. Because deeper models carry a higher risk of overfitting the training data [21], they are also evaluated for their generalizability by testing them on different subjects performing the same task as the training subjects. Aligning with previous research on this subject, a learning rate of 0.0009 was chosen, and the number of epochs was set to 500 [52].

Time series cross-validation is used to evaluate the model's ability to predict torque while fully respecting the temporal order of the data, thereby preventing unrealistic data leakage that would occur if future observations were used to predict the past. To avoid such leakage across training and test sets, for each subject p, movement type o, and MVC level t, the continuous signal is divided into four discrete blocks corresponding to individual contractions. For each fold, the model is trained on all blocks up to a certain block, and tested on the subsequent block, resulting in three folds: Fold 1 trains on block 1 and tests on block 2, Fold 2 trains on blocks 1 and 2 and tests on block 3, and Fold 3 trains on blocks 1 to 3 and tests on block 4. This ensures that information from future contractions does not appear in the training data [48, 24].

To test the generalizability of the model across subjects, the model is trained on data from a subject p performing movement type o at MVC level t, and tested on data from another subject performing the same movement type at the same MVC level. To further investigate transfer learning, the performance of a baseline MLP is compared to that of a pretrained MLP. The pretrained model is first trained on data from subject p performing movement type o at MVC level t, and both the pretrained and baseline models are then evaluated on data from another subject performing the same movement type and task. To determine whether training time is improved, the number of epochs needed to reach a certain loss threshold is evaluated for both models. Additionally, to assess whether the pretrained model requires less training data than the baseline model, its loss is evaluated using smaller portions of training data compared to the baseline.

Chapter 5: Results

The first step was to obtain the raw EMG recordings and torque recordings, yielding $x_{p,q,o,t,i}(k)$ and $y_{p,o,t}(k)$ (see Figure 8.1). The raw EMG recordings were then preprocessed and decomposed into spike trains $\hat{s}_{p,q,o,t}(k)$. Table 8.1 summarizes the average number of motor units decomposed for each muscle for each subject. Since the number of decomposed MUs varied across tasks, the spike trains were summed per muscle to maintain a consistent number of input features for the MLP. These summed spike trains were used to derive muscle-specific feature functions $f_{p,q,o,t}(k)$, as shown in Figure 8.2.
Meanwhile, the torque recordings were filtered and corrected for offset, resulting in $\tilde{y}^{\text{offset}}_{p,o,t}$, as depicted in Figure 8.3. The dataset was then split into blocks for time series cross-validation, and the torque was predicted based on the muscle-specific feature functions, as illustrated in Figure 8.4.

Different MLP architectures were tested, and model performance was evaluated using the $f_{1,\text{Dorsi},30\%}(k)$ features and corresponding labels $\tilde{y}^{\text{offset}}_{1,\text{Dorsi},30\%}(k)$ through 3-fold time series cross-validation. Table 8.2 shows that mean squared error generally decreases as layer width increases, while the number of hidden layers has little effect on the loss.
Notably, the 9-layer architecture with a width of 8 produced substantially higher errors than other configurations. This is likely due to the limited layer width combined with a low learning rate (0.0009), which might have caused the model to converge to a local minimum rather than fully optimizing the loss function. These results highlight the importance of selecting appropriate learning rates, as lower learning rates can lead to stable performance but also increase the risk of getting stuck in local minima [21].

The different MLP architectures were also tested for generalizability. The model was trained using data from subject 1, performing a dorsiflexion movement at 30% MVC, and tested on data from subject 3 performing the same movement type and task.
In contrast to the results in Table 8.2, Table 8.3 reveals an increasing mean squared error as the number of hidden layers and model depth increase. The smaller MLP model with 3 hidden layers and a width of 8 performed significantly better at generalizing to another subject than the larger model with 5 hidden layers and a width of 32. This suggests that smaller models generalize better, possibly because larger models have more parameters, which can lead to overfitting on the training

data and consequently worse generalization [21].

The MLP architecture, consisting of 5 hidden layers with 32 nodes per layer, was used to evaluate performance across MVC tasks, as this model performed best among the tested configurations shown in Table 8.2. The model was trained and evaluated using 3-fold time series cross-validation on data from subject 2 performing plantarflexion movements.
Table 8.4 shows that the mean squared error increases as the MVC level rises. This result can be explained by the principle of *orderly recruitment*, whereby smaller motor units, which generate less force, are recruited first [22]. As torque increases, larger MUs are progressively recruited, resulting in signal overlap with smaller MUs. This increased complexity in HD-sEMG signals at higher MVC levels makes them more difficult to decompose [11, 10, 30], potentially leading to less accurate MU extraction and reduced predictive performance.

The same 5-hidden-layer, 32-nodes-per-layer MLP architecture was also used to evaluate the performance of a pretrained MLP against that of the baseline model. The pretrained model was trained on data from subject 1 performing a plantarflexion movement at 90% MVC, and both the baseline and pretrained models were evaluated on data from subject 2 performing the same task and movement type. Figure 8.5 shows that the pretrained model reached the same level of mean squared error in fewer epochs than the baseline model. This suggests that the computational cost for training to reach the same mean squared error is reduced through transfer learning between subjects performing similar tasks. Moreover. Table 8.5 shows that a pretrained model trained on one block of data performs comparably to the baseline model trained on three blocks of data. This indicates that transfer learning across subjects performing similar tasks reduces the amount of data required to achieve the same level of performance.

Chapter 6: Discussion

Because each trial yields a different number of decomposed motor units per muscle (see Table 8.1), training an MLP directly on individual spike trains and reusing the learned weights across trials was not feasible. To address this, spike trains were summed per muscle, resulting in a fixed number of input features across trials and subjects, which improved generalizability.
However, summing spike trains discards information about individual motor unit activity, potentially reducing the model's ability to capture important distinguishing information. Future studies could explore fitting separate models to each muscle recording site and combining them in a multimodal system. Multimodal systems can leverage information from multiple specialized systems, resulting in improved overall performance [1].

According to Henneman's Size Principle [22], larger motor units produce larger MUAPs than smaller motor units. Fast-ICA is known to be biased towards larger MUAPs [28]. This might have caused the decomposition algorithm to overlook smaller MUs. Future studies could employ a more advanced decomposition algorithm as proposed in Glaser and Holobar [2019] [19], which is better suited for decomposing smaller MUs. This may lead to more accurate spike train decomposition and, in turn, improved prediction performance.

Since HD-sEMG is constrained by the volume conductor effect [15], it cannot capture alpha motor neuron signals, as their action potentials are too weak to be detected at the surface [42]. Invasive EMG, in contrast, offers higher selectivity, enabling more reliable decomposition of individual motor units [31]. Incorporating invasive EMG in future studies could result in more accurate MU decomposition and, consequently, improved torque prediction.

One of the findings suggests that smaller models exhibit improved generalizability, as indicated by a lower mean squared error when evaluated across two different subjects performing the same task. While this outcome may hold in controlled, repetitive clinical settings, it may not extend to more variable or dissimilar movements. Further research could explore transfer learning across subjects performing different tasks to assess the model's robustness under broader conditions.

Normalization is a critical step in machine learning to ensure reliable model performance [9]. However, it must be applied properly to preserve data diversity

and prevent data leakage. To achieve this, the training and test sets must be kept separate and normalized independently, ensuring no information leaks between them [40].
When the training set consists of one subject and the test set of another, normalization can be challenging because each set contains important distinguishing characteristics that can be lost if normalization is not performed carefully. Typically, normalization is applied when input features or labels vary greatly in scale [9].
However, in this thesis, since the muscle-specific feature functions were already within a comparable range (approximately 2 to 7) (see Figure 8.4), normalization was omitted to reduce the risk of data leakage and loss of critical information. This choice might have impacted model performance.

Chapter 7: Conclusion

The Fast-ICA algorithm effectively decomposed motor unit spike trains, enabling the training of a model that learns patterns related to torque production. The results show that smaller MLP architectures perform better at capturing patterns that can be used for generalization to similar tasks performed by different subjects. The findings also indicate that transfer learning can reduce training time and maintain performance with less data. This addresses the challenge of limited data in subject-specific settings and suggests that EMG-driven prosthetics could benefit from shared learned parameters across users. Overall, this approach enables accurate torque prediction by interfacing with the nervous system, supporting the development of leg prostheses that adapt to user intent.

Bibliography

[1] Jean-Baptiste Alayrac et al. *Flamingo: a Visual Language Model for Few-Shot Learning.* 2022. arXiv: 2204.14198 [cs.CV]. URL: https://arxiv.org/abs/2204.14198.

[2] Ahmet Alkan and Mücahid Günay. "Identification of EMG signals using discriminant analysis and SVM classifier". In: *Expert systems with Applications* 39.1 (2012), pp. 44–47.

[3] Amputee Coalition and Avalere Health. *Prevalence of Limb Loss and Limb Difference in the United States: Implications for Public Policy.* White Paper. Based on Avalere Health analysis of insurance claims data. Amputee Coalition, Feb. 2024. URL: https://amputee-coalition.org/wp-content/uploads/Prevalence-of-Limb-Loss-and-Limb-Difference-in-the-United-States_Implications-for-Public-Policy.pdf.

[4] Jaehoon An and Inho Lee. "Artificial neural network-based ground reaction force estimation and learning for dynamic-legged robot systems". In: *PeerJ Computer Science* (2023). DOI: 10.7717/peerj-cs.1720. URL: https://peerj.com/articles/cs-1720/.

[5] Charles Asbury. *Muscle Physiology.* Accessed: 2025-06-13. 2024. URL: https://uw.pressbooks.pub/musclephysiology/chapter/chapter-1/.

[6] Mark F. Bear, Barry W. Connors, and Michael A. Paradiso. *Neuroscience: Exploring the Brain.* 3rd ed. Philadelphia: lippincott williams wilkins Publishers, 2007. ISBN: 978-0781760034.

[7] Asgeir Berland. "A cheap spiking neural model for evolved controllers". Master's thesis. Utrecht University, 2016. URL: https://studenttheses.uu.nl/handle/20.500.12932/23931.

[8] Biodex Medical Systems. *System 4: Instructions for Use (Rev. E).* Accessed: 2025-06-27. Biodex Medical Systems. Shirley, NY, USA, 2024. URL: https://biodexrehab.com/wp-content/uploads/2024/08/20-001-CLR-System-4-IFU-Rev-E.pdf.

[9] Christopher M. Bishop. *Pattern Recognition and Machine Learning.* New York: Springer, 2006.

[10] Marco Carbonaro. "Detecting neuromechanical properties of single motor unit by combining high-density electromyography and ultrafast ultrasonography". Accessed: 2025-05-14. PhD. Thesis. University of Turin, 2023. URL: https://hdl.handle.net/2318/2017723.

[11] Maoqi Chen and Ping Zhou. "High-Density Surface EMG Decomposition: Achievements, Challenges, and Concerns". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 33 (2025), pp. 1212–1219. DOI: `10.1109/TNSRE.2025.3551630`.

[12] Rogerio Cisi and Andre Kohn. "Simulation system of spinal cord motor nuclei and associated nerves and muscles, in a Web-based architecture". In: *Journal of computational neuroscience* 25 (June 2008), pp. 520–42. DOI: `10.1007/s10827-008-0092-8`.

[13] H. Peter Clamann. "Motor Units and Their Activity during Movement". In: *Motor Coordination*. Ed. by Arnold L. Towe and Erich S. Luschei. Boston, MA: Springer US, 1981, pp. 69–92. ISBN: 978-1-4684-3884-0. DOI: `10.1007/978-1-4684-3884-0_2`. URL: `https://doi.org/10.1007/978-1-4684-3884-0_2`.

[14] B.J. van Dieren. *Decoding Motor Unit Firing Events During Walking : An Adaptive Approach Validated with Intramuscular Electromyography*. Mar. 2025. URL: `http://essay.utwente.nl/106028/`.

[15] Dario Farina and Aleš Holobar. "Characterization of Human Motor Units From Surface EMG Decomposition". In: *Proceedings of the IEEE* 104.2 (2016), pp. 353–373. DOI: `10.1109/JPROC.2015.2498665`.

[16] Dario Farina et al. "Man/machine interface based on the discharge timings of spinal motor neurons after targeted muscle reinnervation". In: *Nature biomedical engineering* 1.2 (2017), p. 0025.

[17] Aaron Fleming et al. "Myoelectric control of robotic lower limb prostheses: a review of electromyography interfaces, control paradigms, challenges and future directions". In: *Journal of Neural Engineering* 18.4 (2021), p. 041004. DOI: `10.1088/1741-2552/ac1176`.

[18] Weidong Geng et al. "Gesture recognition by instantaneous surface EMG images". In: *Scientific reports* 6.1 (2016), p. 36571.

[19] Vojko Glaser and Aleš Holobar. "Motor Unit Identification From High-Density Surface Electromyograms in Repeated Dynamic Muscle Contractions". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 27.1 (2019), pp. 66–75. DOI: `10.1109/TNSRE.2018.2885283`.

[20] Antonio Gogeascoechea et al. "Characterization of motor unit firing and twitch properties for decoding musculoskeletal force in the human ankle joint in vivo". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 31 (2023), pp. 4040–4050.

[21] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. `http://www.deeplearningbook.org`. MIT Press, 2016.

[22] Elwood Henneman. “Relation between Size of Neurons and Their Susceptibility to Discharge”. In: *Science* 126.3287 (1957). Accessed via University of Maryland proxy, pp. 1345–1347. URL: http://www.jstor.org/stable/1752769.

[23] Ales Holobar and Damjan Zazula. “Multichannel Blind Source Separation Using Convolution Kernel Compensation”. In: *IEEE Transactions on Signal Processing* 55.9 (2007), pp. 4487–4496. DOI: 10.1109/TSP.2007.896108.

[24] Rob J. Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. 3rd. Accessed on 2025-07-02. Melbourne, Australia: OTexts, 2021. URL: https://otexts.com/fpp3/index.html.

[25] Pidikiti Keerthi, Srisai Satya Santosh Y, and Iniyan S. “Lung Cancer Classification Using Deep Neural Networks”. In: *2025 AI-Driven Smart Healthcare for Society 5.0*. 2025, pp. 1–6. DOI: 10.1109/IEEECONF64992.2025.10963162.

[26] Łukasz Kidziński et al. “Artificial intelligence for prosthetics: Challenge solutions”. In: *The NeurIPS’18 Competition: From Machine Learning to Intelligent Conversations*. Springer. 2020, pp. 69–128.

[27] S Krishnapriya, Jaya Prakash Sahoo, and Samit Ari. “Surface electromyography based hand gesture signal classification using 1d cnn”. In: *2023 International Conference on Intelligent Systems, Advanced Computing and Communication (ISACC)*. IEEE. 2023, pp. 1–6.

[28] Logan Patrick Leahy. “Estimating output torque via amplitude estimation and neural drive: a high-density sEMG study”. Master’s thesis. Massachusetts Institute of Technology, Department of Mechanical Engineering, 2020. URL: https://dspace.mit.edu/handle/1721.1/127134.

[29] Diu Khue Luu et al. “Artificial intelligence enables real-time and intuitive control of prostheses via nerve interface”. In: *IEEE Transactions on Biomedical Engineering* 69.10 (2022), pp. 3051–3063.

[30] Roberto Merletti. *Electromyography : Physiology, Engineering, and Non-Invasive Applications*. Vol. 11. Aug. 2004. ISBN: 0471675806. DOI: 10.1002/0471678384.

[31] Roberto Merletti and Dario Farina. “Analysis of Intramuscular electromyogram signals”. In: *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences* 367 (Nov. 2008), pp. 357–68. DOI: 10.1098/rsta.2008.0235.

[32] Francesco Negro et al. “Multi-channel intramuscular and surface EMG decomposition by convolutive blind source separation”. In: *Journal of neural engineering* 13.2 (2016), p. 026027.

[33] Rafael Ornelas-Kobayashi, Rienke Mooiweer, and M Sartori. “Towards Personalized Motor-Restoring Technologies: Characterizing Neural Data-Driven Alpha-Motoneuron Model Parameters”. In: *2024 10th IEEE RAS/EMBS International Conference for Biomedical Robotics and Biomechatronics (BioRob)*. IEEE. 2024, pp. 1581–1586.

[34] Sinno Jialin Pan and Qiang Yang. “A Survey on Transfer Learning”. In: *IEEE Transactions on Knowledge and Data Engineering* 22.10 (2010), pp. 1345–1359. DOI: 10.1109/TKDE.2009.191.

[35] John G. Proakis and Dimitris G. Manolakis. *Digital Signal Processing: Principles, Algorithms, and Applications*. 4th. Upper Saddle River, NJ: Pearson Prentice Hall, 2006.

[36] Nicola Rieke et al. “The future of digital health with federated learning”. In: *npj Digital Medicine* 3 (Dec. 2020). DOI: 10.1038/s41746-020-00323-1.

[37] Jason A. Rivera et al. “Estimating Recent US Limb Loss Prevalence and Updating Future Projections”. In: *Archives of Rehabilitation Research and Clinical Translation* 6.4 (2024), p. 100376. DOI: 10.1016/j.arrct.2024.100376. URL: https://www.archives-rrct.org/article/S2590-1095(24)00089-2/fulltext.

[38] Frank Rosenblatt. *The Perceptron: A Theory of Statistical Separability in Cognitive Systems (Project PARA)*. Technical Report VG-1196-G-1. Buffalo, NY: Cornell Aeronautical Laboratory, 1958. URL: https://books.google.com/books/about/The_Perceptron.html?id=7Q4-AQAAIAAJ.

[39] Massimo Sartori, Utku Yavuz, and Dario Farina. “In Vivo Neuromechanics: Decoding Causal Motor Neuron Behavior with Resulting Musculoskeletal Function”. In: *Scientific Reports* 7 (Oct. 2017). DOI: 10.1038/s41598-017-13766-6.

[40] Scikit-learn developers. *Common pitfalls in machine learning*. Accessed: 2025-06-27. URL: https://scikit-learn.org/stable/common_pitfalls.html#.

[41] Jost Tobias Springenberg et al. *Striving for Simplicity: The All Convolutional Net*. 2015. arXiv: 1412.6806 [cs.LG]. URL: https://arxiv.org/abs/1412.6806.

[42] Marcus Talke. “Using Supervised Deep Learning for Real-Time Motor Unit Decomposition of High-Density Surface Electromyography Signals”. Accessed: 2025-05-14. Master’s thesis. Espoo, Finland: Aalto University, June 2022. URL: https://aaltodoc.aalto.fi/items/65be6b77-9880-4f05-a9fe-26dd162e577b.

[43] J. Thomas, Y. Deville, and S. Hosseini. “Time-domain fast fixed-point algorithms for convolutive ICA”. In: *IEEE Signal Processing Letters* 13.4 (2006), pp. 228–231. DOI: 10.1109/LSP.2005.863638.

[44] Arnold L. Towe and Erich S. Luschei. *Motor Coordination.* Vol. 5. Handbook of Behavioral Neurobiology. First edition. New York, NY: Plenum Press, 1981. ISBN: 0306406136, 978-0306406133. URL: https://link.springer.com/book/10.1007/978-1-4684-3884-0.

[45] University of Texas Health Science Center at Houston. *Motor Units and Muscle Receptors.* https://nba.uth.tmc.edu/neuroscience/m/s3/chapter01.html. Accessed: 2025-04-19.

[46] Pauli Virtanen et al. “SciPy 1.0: Fundamental algorithms for scientific computing in Python”. In: *Nature Methods* 17.3 (2020), pp. 261–272.

[47] World Medical Association. “World Medical Association Declaration of Helsinki: Ethical principles for medical research involving human subjects”. In: *JAMA* 310.20 (2013), pp. 2191–2194. DOI: 10.1001/jama.2013.281053. URL: https://jamanetwork.com/journals/jama/fullarticle/1760318.

[48] Jehan Yang et al. *EMGBench: Benchmarking Out-of-Distribution Generalization and Adaptation for Electromyography.* 2024. arXiv: 2410.23625 [cs.LG]. URL: https://arxiv.org/abs/2410.23625.

[49] Jason Yosinski et al. “How transferable are features in deep neural networks?” In: *CoRR* abs/1411.1792 (2014). arXiv: 1411.1792. URL: http://arxiv.org/abs/1411.1792.

[50] Xiaolong Zhai et al. “Self-recalibrating surface EMG pattern recognition for neuroprosthesis control based on convolutional neural network”. In: *Frontiers in neuroscience* 11 (2017), p. 379.

[51] Daohui Zhang et al. “PCA and LDA for EMG-based control of bionic mechanical hand”. In: *2012 IEEE international conference on information and automation.* IEEE. 2012, pp. 960–965.

[52] Longbin Zhang et al. “Ankle Joint Torque Estimation Using an EMG-Driven Neuromusculoskeletal Model and an Artificial Neural Network Model”. In: *IEEE Transactions on Automation Science and Engineering* 18.2 (2021), pp. 564–573. DOI: 10.1109/TASE.2020.3033664.

[53] H Ziai and C Menon. “Comparison of regression models for estimation of isometric wrist joint torques using surface electromyography”. In: *Journal of NeuroEngineering and Rehabilitation* (2011). DOI: 10.1186/1743-0003-8-56.

Chapter 8: Appendix

Algorithm 8.1 Fast-ICA algorithm [32]

```
    Subtract the mean from the observations $\mathbf{X}$
    Whiten $\mathbf{X}$
    Initialize the matrix $\mathbf{B}$ to empty matrix
    $g(x) = x^3$
    $g'(x) = 3x^2$
 1: procedure FAST ICA($\mathbf{X}, M, L$)
 2:     for $i = 1, 2, ..., M$ do
 3:         Initialize the vector $w_i(0)$ and $w_i(-1)$
 4:         while $|w_i(t)^T w_i(t-1) - 1| < \text{TOLx}$ do
 5:             Fixed-Point Algorithm:
                    $$w_i(t) = \mathbb{E}[z \cdot g(w_i(t-1)^T \cdot z)] - A \cdot w_i(t-1)$$
                    $$\text{with } A = \mathbb{E}[g'(w_i(t-1)^T z)]$$
 6:             Orthogonalization:
                    $$w_i(t) = w_i(t) - \mathbf{B}\mathbf{B}^T w_i(t)$$
 7:             Normalization:
                    $$w_i(t) = \frac{w_i(t)}{||w_i(t)||}$$
 8:         end while
 9:         if SIL $> 0.9$ then
10:             Accept the source estimate
11:             Add $w_i$ to the matrix $B$
12:         end if
13:     end for
14: end procedure
```

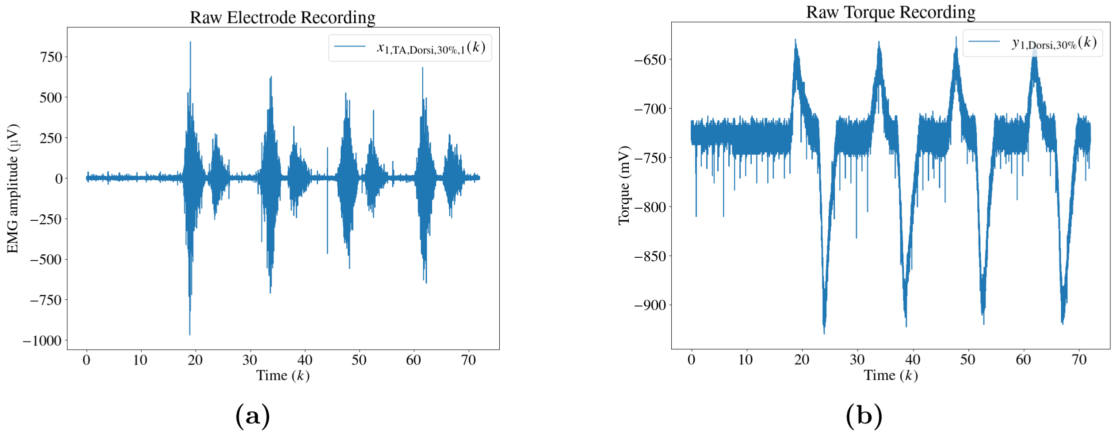


Figure 8.1: Raw EMG (a) and torque (b) recordings from subject 1 performing four repeated dorsiflexion movements at 30% MVC, recorded from the first electrode over the tibialis anterior muscle.

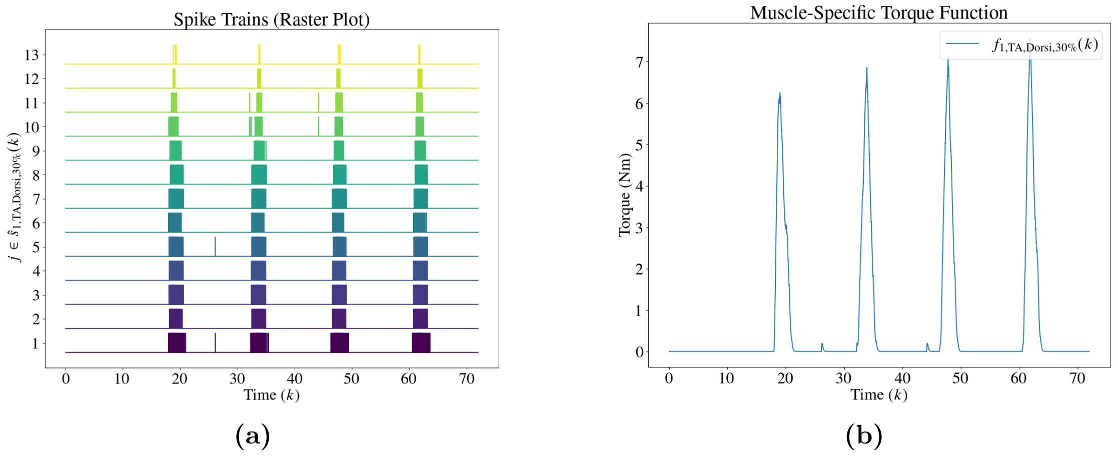
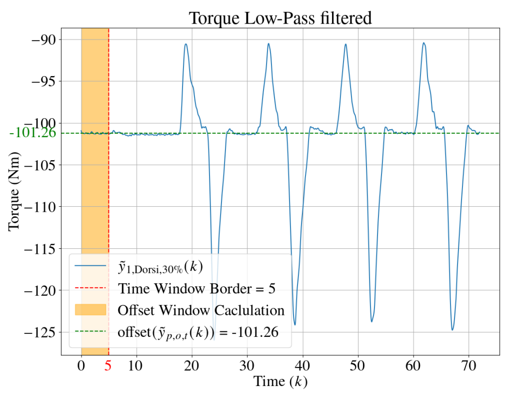
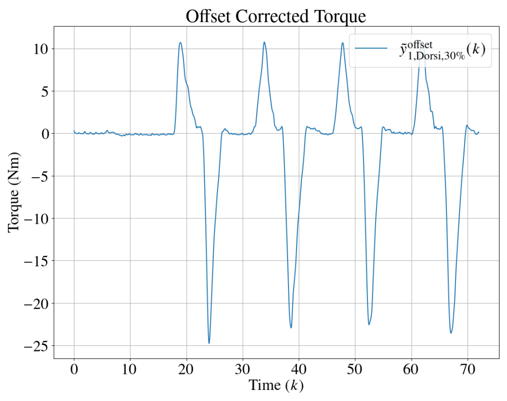


Figure 8.2: Spike trains (sources) (a) and muscle-specific feature function (b) obtained from the tibialis anterior muscle of subject 1 during four repeated dorsiflexion movement at 30% MVC.

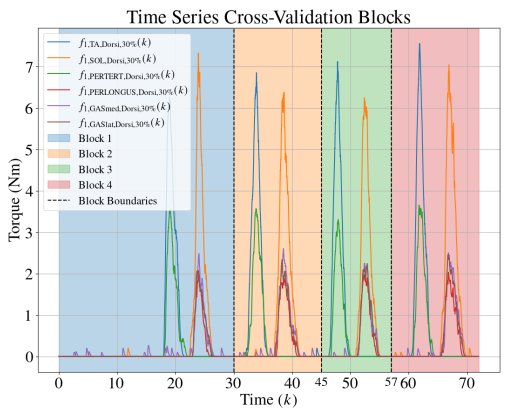


(a)

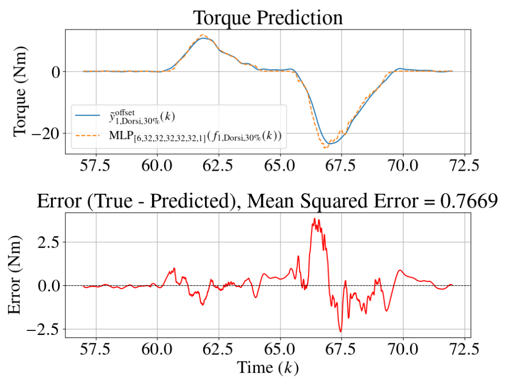


(b)

Figure 8.3: The "Offset Window Calculation" in (a) Displays the 5-second window which is used to calculate offset($\tilde{y}_{1,\text{Dorsi},30\%}(k)$ (see Equations 4.6). This offset is used to center the torque around zero (b). From subject 1 during four repeated dorsiflexion movements at 30% MVC.



(a)



(b)

Figure 8.4: (a) Muscle-specific feature functions and the four blocks data are split into, each corresponding to individual contractions, for time-series cross-validation. (b) Visualization of the test set performance in the final fold, where the model is trained on data from blocks 1-3, and tested on block 4. The plot shows predicted torque alongside true torque of subject 1 during a dorsiflexion movement at 30% MVC.

Subject	TA	SOL	PT	PL	GM	GL
Subject 1	14.9 ± 3.2	19.9 ± 3.7	1.2 ± 1.6	3.1 ± 2.7	6.8 ± 3.3	4.2 ± 3.2
Subject 2	15.8 ± 4.4	9.6 ± 2.7	0.1 ± 0.4	6.9 ± 3.6	12.9 ± 6.7	5.1 ± 2.1
Subject 3	20.5 ± 2.0	24.8 ± 2.7	0.4 ± 0.5	5.9 ± 1.4	9.0 ± 4.8	7.8 ± 3.9

Table 8.1: Average number of motor units decomposed (± standard deviation) per muscle and per subject, averaged over all tasks and movement types.

Hidden Layers	Width 8	Width 16	Width 32
3	0.81	0.76	0.74
5	0.80	0.81	0.70
9	7.12	0.84	0.77

Table 8.2: Average 3-fold time series cross-validation loss (mean squared error) for MLP architectures trained and tested on subject 1, performing a dorsiflexion movement at 30% MVC. The architectures used a consistent number of nodes per hidden layer; for example, an MLP with 5 hidden layers and width 8 corresponds to the structure $\text{MLP}_{6\times8\times8\times8\times8\times8\times1}$.

Hidden Layers	Width 8	Width 16	Width 32
3	1.18	2.13	8.36
5	2.47	2.66	31.40
9	5.46	6.09	25.81

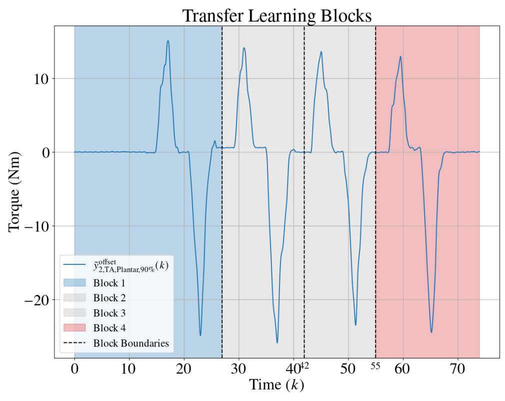
Table 8.3: Mean squared error for MLP architectures trained on subject 1 and tested on subject 3, both performing the same dorsiflexion movement at 30% MVC. The architectures used a consistent number of nodes per hidden layer; for example, an MLP with 5 hidden layers and width 8 corresponds to the structure $\text{MLP}_{6\times8\times8\times8\times8\times8\times1}$.

Tasks	Plantarflexion
30% MVC	0.30
50% MVC	1.61
70% MVC	2.33
90% MVC	5.06

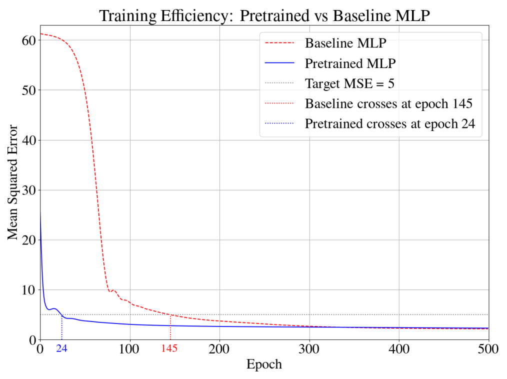
Table 8.4: Average 3-fold time series cross-validation loss (mean squared error) on data from subject 2 performing plantarflexion at 30%, 50%, 70%, and 90% MVC. The MLP architecture consisted of 5 hidden layers and 32 nodes per layer.

Training Blocks	Pretrained	Test Block	MSE
Block 1	No	Block 4	8.67
Block 1	Yes	Block 4	5.94
Blocks 1-3	No	Block 4	5.74
Blocks 1-3	Yes	Block 4	6.17

Table 8.5: Mean squared error obtained using a baseline and a pretrained MLP. The pretrained model was first trained on data from subject 1 performing a plantarflexion movement at 90% MVC. Both the pretrained and baseline models were then trained on data from subject 2, performing the same plantarflexion movement at 90% MVC. Both models used an architecture with 5 hidden layers and 32 nodes per layer.



(a)



(b)

Figure 8.5: Data from subject 2 performing a plantarflexion movement at 90% MVC. (a) Overview of the blocks used for evaluating transfer learning, showing the offset-corrected torque from subject 2. (b) Mean squared error loss over training epochs comparing the baseline and pretrained MLP models on data of subject 2. The pretrained MLP was first trained on data from subject 1 performing a plantarflexion movement at 90% MVC. Both the pretrained and baseline models were then trained on data from subject 2, performing the same plantarflexion movement at 90% MVC. Both models used an architecture with 5 hidden layers and 32 nodes per layer.