On the Utility of Representation Learning Algorithms for Myoelectric Interfacing

Olsson, Alexander

2023

Document Version:
Publisher's PDF, also known as Version of record

On the Utility of Representation Learning Algorithms for Myoelectric Interfacing

Alexander E. Olsson

LUND
UNIVERSITY

Doctoral Dissertation

by due permission of the Faculty of Engineering, Lund University, Sweden.
To be defended in E:B, Ole Römers väg 3, Lund.
June 9, 2023 at 09:15

Faculty opponent
Associate Professor Erik Scheme

Organization
LUND UNIVERSITY

Department of Biomedical Engineering
P.O. Box 118
SE–221 00 LUND, Sweden

Document name
DOCTORAL DISSERTATION

Date of disputation
2023-06-09

Sponsoring organization
The Promobilia Foundation

Author(s)
Alexander E. Olsson

Title and subtitle
On the utility of representation learning algorithms for myoelectric interfacing

Abstract
Electrical activity produced by muscles during voluntary movement is a reflection of the firing patterns of relevant motor neurons and, by extension, the latent motor intent driving the movement. Once transduced via electromyography (EMG) and converted into digital form, this activity can be processed to provide an estimate of the original motor intent and is as such a feasible basis for non-invasive efferent neural interfacing. EMG-based motor intent decoding has so far received the most attention in the field of upper-limb prosthetics, where alternative means of interfacing are scarce and the utility of better control apparent. Whereas myoelectric prostheses have been available since the 1960s, available EMG control interfaces still lag behind the mechanical capabilities of the artificial limbs they are intended to steer—a gap at least partially due to limitations in current methods for translating EMG into appropriate motion commands. As the relationship between EMG signals and concurrent effector kinematics is highly non-linear and apparently stochastic, finding ways to accurately extract and combine relevant information from across electrode sites is still an active area of inquiry.

This dissertation comprises an introduction and eight papers that explore issues afflicting the status quo of myoelectric decoding and possible solutions, all related through their use of learning algorithms and deep Artificial Neural Network (ANN) models. Paper I presents a Convolutional Neural Network (CNN) for multi-label movement decoding of high-density surface EMG (HD-sEMG) signals. Inspired by the successful use of CNNs in Paper I and the work of others, Paper II presents a method for automatic design of CNN architectures for use in myocontrol. Paper III introduces an ANN architecture with an appertaining training framework from which simultaneous and proportional control emerges. Paper IV introduce a dataset of HD-sEMG signals for use with learning algorithms. Paper V applies a Recurrent Neural Network (RNN) model to decode finger forces from intramuscular EMG. Paper VI introduces a Transformer model for myoelectric interfacing that do not need additional training data to function with previously unseen users. Paper VII compares the performance of a Long Short-Term Memory (LSTM) network to that of classical pattern recognition algorithms. Lastly, paper VIII describes a framework for synthesizing EMG from multi-articulate gestures intended to reduce training burden.

Key words
EMG, Machine Learning, Deep Learning, Feature Learning, Biosignal Processing, Gesture Recognition, Muscle-Computer Interfaces, Myoelectric Control, Upper Limb Prosthetics

Classification system and/or index terms (if any)

Supplementary bibliographical information
ISRN: LUTEDX/LUTEDX/TEEM-1134-SE, Report No. 2/23

Language
English

ISSN and key title

ISBN
978-91-8039-736-0 (print)
978-91-8039-735-3 (electronic)

Recipient's notes

Number of pages
283

Price

Security classification

Signature _____ Date _____

To Scronkfinkle the Sparrow.

All stable processes we shall predict.
All unstable processes we shall control.
—John von Neumann

Public defence
June 9, 2023 at 09:15 in E:B, E-building, LTH, Ole Römers väg 3, 223 63 Lund, Sweden

Supervisors
Accociate Professor Christian Antfolk
Dr. Nebojša Malešević
Department of Biomedical Engineering, Faculty of Engineering, Lund University, Lund, Sweden

Professor Anders Björkman
Department of Hand Surgery, Institute of Clinical Sciences, Sahlgrenska Academy, University of Gothenburg, Sahlgrenska University Hospital, Sweden

Faculty opponent
Associate Professor Erik Scheme
Department of Electrical and Computer Engineering, University of New Brunswick, Fredericton, Canada

Examination board
Associate Professor Magnus Jonsson
Department of Computer Sciene and Media Technology, Malmö University, Malmö, Sweden

Assistant Professor Silvia Muceli
Department of Electrical Engineering, Chalmers University of Technology, Gothenburg, Sweden

Assistant Professor Ruoli Wang
Department of Engineering Mechanics, KTH Royal Institute of Technology, Stockholm, Sweden

Deputy member: Associate Professor Kristian Soltesz
Department of Automatic Control, Faculty of Engineering, Lund University, Lund, Sweden

Chairman
Associate Professor Johan Nilsson
Department of Biomedical Engineering, Faculty of Engineering, Lund University, Lund, Sweden

Cover illustration
Stylized rendition of an artificial neural network processing myoelectric signals from the upper limbs. Created using *Stable Diffusion 1.5*, a machine learning model for image synthesis.

ISBN: 978-91-8039-736-0 (printed version)
ISBN: 978-91-8039-735-3 (electronic version)
Report No. 2/23
ISRN: LUTEDX/TEEM-1134-SE
Printed by Tryckeriet, E-building, Faculty of Engineering, Lund University, Lund Sweden

Contents

Populärvetenskaplig Sammanfattning

När vi människor försöker röra på oss skickas elektriska impulser genom nervsystemet från hjärnan ner till de muskler som är inblandade i den tilltänkta rörelsen. Att kunna mäta sådana nervimpulser hade varit av värde för en rad tillämpningar—information hade då kunnat överföras direkt från nervsystemet utan att gå via långsamma, mekaniska gränssnitt så som tangentbord och pekskärmar. Olyckligtvis är det i dagsläget både svårt och riskfyllt att mäta nervimpulser direkt. Motoriska nerver är små och omgivna av vävnad och kräver därför svåra kirurgiska ingrepp för att nås.

Ett långt mer praktiskt alternativ är att istället utnyttja den elektriska aktivitet som uppstår i själva musklerna när nervinpulserna når fram dit. På grund av sin biokemiska sammansättning genomgår muskelfibrer under sammandragning elektriska urladdningar som till sin natur är mycket lika de som nervcellerna själva genomgår. Eftersom muskler är mycket större än nerver uppstår elektriska signaler som är mycket starkare än de bakomliggande nervimpulserna—muskelgenererad elektrisk aktivitet i till exempel underarmen är kraftfull nog att mätas från huden. En teknik som kan användas för att mäta aktiviteten kallas yt-elektromyografi, förkortat EMG, och är typsikt inte mer krånglig än att elektroder (egentligen bara små bitar metall) placeras intill muskeln eller musklerna som ska mätas. Signaler som samlas in på detta vis kan, efter lämplig behandling, i teorin användas för att erhålla den avsikt som låg bakom rörelsen. Begreppet ”i teorin” utför mycket av arbetet i föregående mening; sambandet mellan muskelsignaler och rörelseavsikt är mycket komplicerat och kräver särskilda algoritmer för att avkodas. Denna avhandling handlar om sådana algoritmer.

Ett välkänt användningsområde för metoder som härleder rörelseavsikt från muskelsignaler är protesstyrning. Bland de muskler som styr handen är flertalet belägna i underarmen, så vid handamputation är det sannolikt att muskler bevaras och utgör förstärkare av de nervimpulser som inte längre resulterar i rörelse. Så kallade *myoelektriska* proteser kan styras av signaler från sådana muskler. Trots att tekniken funnits länge har denna typ av protes relativt svagt genomslag, och många amputerade väljer att inte använda sin protes alls. Att styrningen inte är naturlig eller exakt nog uppges ofta som ett skäl till missnöje. Detta kan jämföras med samtida protesers mekaniska förmågor, vilka i mångt och mycket motsvarar den naturliga handens.

Arbetet som presenteras i denna avhandling handlar om att utveckla och förbättra metoder som kan användas för att extrahera rörelseavsikt från muskelsignaler, med protesstyrning och andra typer av människa-datorinteraktion som möjliga tillämpningar. Konkret presenteras ett antal *inlärningsalgoritmer* för detta ändamål, förenade i att de helt eller delvis använder sig utav djupa *artificiella neuronnät*. Sådana algoritmer kan lära sig att känna igen mönster i muskelsignalerna som är specifika för vissa rörelser och använda denna information för att sluta sig till användarens avsikt. På sådant vis behöver inte det komplicerade sambandet mellan avsikt och muskelaktivitet modelleras explicit; det kan istället läras från exempel (så kallad träningsdata) i form av muskelsignaler som samlas in från den tilltänka användaren. Ett antal fördelar särskiljer de förslag som presenteras i denna avhandlings 8 artiklar från tidigare forskning inom samma fält. Som exempel presenteras metoder som tillåter *samtidig* styrning av flera frihetsgrader (t. ex. enskilda fingrar). Vidare presenteras en metod som möjliggör styrning som är *proportionell* mot den underliggande muskelaktivitetens intensitet, trots att träningsdatan bara behöver samlas in under en kontraktionsstyrka. Det introduceras häri även en neuronnätsmodell som kan avkoda rörelseavsikt direkt från nya användare utan behov att samla in ny träningsdata. Sådana egenskaper är tänkta att tillåta framtida gränssnitt mellan muskler och externa enheter som är både mer exakta och mer naturliga än de alternativ som finns tillgängliga idag.

Abstract

Electrical activity produced by muscles during voluntary movement is a reflection of the firing patterns of relevant motor neurons and, by extension, the latent motor intent driving the movement. Once transduced via electromyography (EMG) and converted into digital form, this activity can be processed to provide an estimate of the original motor intent and is as such a feasible basis for non-invasive efferent neural interfacing. EMG-based motor intent decoding has so far received the most attention in the field of upper-limb prosthetics, where alternative means of interfacing are scarce and the utility of better control apparent. Whereas myoelectric prostheses have been available since the 1960s, available EMG control interfaces still lag behind the mechanical capabilities of the artificial limbs they are intended to steer—a gap at least partially due to limitations in current methods for translating EMG into appropriate motion commands. As the relationship between EMG signals and concurrent effector kinematics is highly non-linear and apparently stochastic, finding ways to accurately extract and combine relevant information from across electrode sites is still an active area of inquiry.

This dissertation comprises an introduction and eight papers that explore issues afflicting the status quo of myoelectric decoding and possible solutions, all related through their use of learning algorithms and deep Artificial Neural Network (ANN) models. Paper I presents a Convolutional Neural Network (CNN) for multi-label movement decoding of high-density surface EMG (HD-sEMG) signals. Inspired by the successful use of CNNs in Paper I and the work of others, Paper II presents a method for automatic design of CNN architectures for use in myocontrol. Paper III introduces an ANN architecture with an appertaining training framework from which simultaneous and proportional control emerges. Paper IV introduce a dataset of HD-sEMG signals for use with learning algorithms. Paper V applies a Recurrent Neural Network (RNN) model to decode finger forces from intramuscular EMG. Paper VI introduces a Transformer model for myoelectric interfacing that do not need additional training data to function with previously unseen users. Paper VII compares the performance of a Long Short-Term Memory (LSTM) network to that of classical pattern recognition algorithms. Lastly, paper VIII describes a framework for synthesizing EMG from multi-articulate gestures intended to reduce training burden.

List of Papers

Journal Papers

I. **Extraction of multi-labelled movement information from the raw HD-sEMG image with time-domain depth**
A. E. Olsson, P. Sager, E. Andersson, A. Björkman, N. Malešević, C. Antfolk
Published in: Scientific Reports 9, 7244 (2019)

Author's contributions: Part of study design, design and implementation of algorithms, analysis of results, manuscript writing.

II. **Automatic discovery of resource-restricted Convolutional Neural Network topologies for myoelectric pattern recognition**
A. E. Olsson, A. Björkman, C. Antfolk
Published in: Computers in Biology and Medicine 120, 103723 (2020)

Author's contributions: Study design, design and implementation of algorithms, analysis of results, manuscript writing.

III. **Learning regularized representations of categorically labelled surface EMG enables simultaneous and proportional myoelectric control**
A. E. Olsson, N. Malešević, A. Björkman, C. Antfolk
Published in: Journal of NeuroEngineering and Rehabilitation 18, 35 (2021)

Author's contributions: Study design, data collection, design and implementation of algorithms, analysis of results, manuscript writing.

IV. **A database of high-density surface electromyogram signals comprising 65 isometric hand gestures**
N. Malešević, **A. Olsson**, P. Sager, E. Andersson, C. Cipriani, M. Controzzi, A. Björkman, C. Antfolk
Published in: Scientific Data 8, 63 (2021)

Author's contributions: Part of data collection, part of analysis of results, part of the manuscript writing.

v. **End-to-end estimation of hand-and wrist forces from raw intramuscular EMG signals using LSTM networks**
A. E. Olsson, N. Malešević, A. Björkman, C. Antfolk
Published in: Frontiers in Neuroscience 15, 777329 (2021)
Author's contributions: Study design, design and implementation of algorithms, analysis of results, manuscript writing.

Manuscripts

vi. **Calibration-free myoelectric decoding**

A. E. Olsson, N. Malešević, A. Björkman, C. Antfolk
Manuscript
Author's contributions: Study design, design and implementation of algorithms, analysis of results, manuscript writing.

Conference Papers

vii. **Exploiting the intertemporal structure of the upper-limb sEMG: comparisons between an LSTM network and cross-sectional myoelectric pattern recognition methods**
A. Olsson, N. Malešević, A. Björkman, C. Antfolk
In *41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC) 2019*
Author's contributions: Study design, design and implementation of algorithms, analysis of results, manuscript writing.

viii. **Can deep synthesis of EMG overcome the geometric growth of training data required to recognize multiarticulate motions?**
A. E. Olsson, N. Malešević, A. Björkman, C. Antfolk
In *43rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC) 2021*
Author's contributions: Study design, design and implementation of algorithms, analysis of results, manuscript writing.

Not Included

1. **An Algorithm Calibrated with Categorically Labelled EMG for End-to-End Estimation of Continuous Hand Kinematics**
   A. E. Olsson, N. Malešević, Anders Björkman, Christian Antfolk
   In *Myoelectric Controls Symposium (MEC) 2020*

2. **Exhaustive Search on Graphics Hardware Enables Fast Selection of Optimal sEMG Channels and Signal Features for Myoelectric Pattern Recognition**
   A. E. Olsson, N. Malešević, Anders Björkman, Christian Antfolk
   Abstract presented as poster at the *International Neurorehabilitation Symposium (INRS) 2022*

Notation

Comments

Indexing starts at 1 for all dimensions of vectors, matrices and higher-order tensors. For data tensors, the first dimension (row) indexes examples and subsequent dimensions (columns and higher-dimensional equivalents) index features. Except for transpose and inverse operators, superscripts are reserved for labelling (e.g. $\boldsymbol{X}^{Tr}$ and $\boldsymbol{X}^{Val}$ refers to training- and validation-data, respectively), whereas exponentiation is denoted by ^. Division with non-scalar operands is always performed element-wise.

Sets

$\mathbb{A}$	The set $\mathbb{A}$.
$\mathbb{A}^N$	The N-ary Cartesian power over $\mathbb{A}$.
$\mathbb{A}^{M\times N}$	The N-ary Cartesian power over $\mathbb{A}^M$.
$\mathbb{R}$	The set of real numbers.
$\{1, ..., c\}$	The set containing all integers from 1 to c.

Quantities & Indexing

$a \in \mathbb{R}$	The scalar value a.
$\boldsymbol{a} \in \mathbb{R}^N$	The N-dimensional vector $\boldsymbol{a}$.
$a_j \in \mathbb{R}$	The jth entry of vector $\boldsymbol{a}$.
$\boldsymbol{A} \in \mathbb{R}^{N\times M}$	The matrix $\boldsymbol{A}$ with N rows and M columns.
$A_{i,j} \in \mathbb{R}$	The entry of matrix $\boldsymbol{A}$ at row i and column j.
$\boldsymbol{A}_{j,*} \in \mathbb{R}^M$	The jth row of matrix $\boldsymbol{A} \in \mathbb{R}^{N\times M}$.
$\boldsymbol{A}_{*,j} \in \mathbb{R}^N$	The jth column of matrix $\boldsymbol{A} \in \mathbb{R}^{N\times M}$.

Operators

$\boldsymbol{a} \odot \boldsymbol{b} \in \mathbb{R}^N$	Element-wise product of vectors $\boldsymbol{a} \in \mathbb{R}^N$ and $\boldsymbol{b} \in \mathbb{R}^N$.
$\boldsymbol{a} \cdot \boldsymbol{b} \in \mathbb{R}$	Scalar product of vectors $\boldsymbol{a} \in \mathbb{R}^N$ and $\boldsymbol{b} \in \mathbb{R}^N$.
$\boldsymbol{A} \cdot \boldsymbol{a} \in \mathbb{R}^M$	Product of matrix $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ and vector $\boldsymbol{a} \in \mathbb{R}^N$.
$\boldsymbol{A} \cdot \boldsymbol{B} \in \mathbb{R}^{M \times N}$	Product of matrix $\boldsymbol{A} \in \mathbb{R}^{M \times K}$ and matrix $\boldsymbol{B} \in \mathbb{R}^{K \times N}$.
$\boldsymbol{A}^{-1} \in \mathbb{R}^{N \times N}$	Inverse of matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$.
$\boldsymbol{A}^T \in \mathbb{R}^{M \times N}$	Transpose of matrix $\boldsymbol{A} \in \mathbb{R}^{N \times M}$.
$\det(\boldsymbol{A}) \in \mathbb{R}$	Determinant of matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$
$\left.\frac{\partial f(x_1, x_2, \ldots)}{\partial x_j}\right\|_{x_j = a} \in \mathbb{R}$	Partial derivative of function f w.r.t. x evaluated at $x_j = a$
$\nabla f(\boldsymbol{x})$	Gradient of scalar-valued function $f\colon \mathbb{R}^N \to \mathbb{R}$.
$\nabla_{\boldsymbol{a}} f(\boldsymbol{a}, \boldsymbol{b}, \ldots)$	Gradient of scalar-valued function f w.r.t. argument $\boldsymbol{a}$.

Functions

$f\colon \mathbb{A} \to \mathbb{B}$	Function f with domain $\mathbb{A}$ and range $\mathbb{B}$.
$f(\boldsymbol{x}; \boldsymbol{\theta})$	Function of $\boldsymbol{x}$ parametrized by $\boldsymbol{\theta}$.
$H(x) = \begin{cases} 1 \text{ if } x \geq 0 \\ 0 \text{ if } x < 0 \end{cases}$	The Heaviside step function.
$sgn(x) = \begin{cases} 1 \text{ if } x > 0 \\ 0 \text{ if } x = 0 \\ -1 \text{ if } x < 0 \end{cases}$	The sign function.
$\mathbb{I}(C) = \begin{cases} 1 \text{ if } C \text{ is true} \\ 0 \text{ if } C \text{ is false} \end{cases}$	The logical indicator function.

Acronyms & Abbreviations

α-**MN** Alpha Motor Neuron

ANN Artificial Neural Network

AP Action Potential

CNN Convolutional Neural Network

CNS Central Nervous System

DL Deep Learning

DoF Degree of Freedom

EEG Electroencephalography

EMG Electromyography

FNIRS Functional Near-Infrared Spectroscopy

HD-sEMG High-Density Surface Electromyography

LDA Linear Discriminant Analysis

LSTM Long Short-Term Memory

MAV Mean Absolute Value

MCI Muscle-Computer Interface

ML Machine Learning

MLP Multi-Layer Perceptron

MU Motor Unit

MUAP Motor Unit Action Potential

NMF Non-Negative Matrix Factorization

NMJ Neuromuscular Junction

PCA Principal Component Analysis

PMC Premotor Cortex

RMSE Root Mean Squared Error

RNN Recurrent Neural Network

SMA Supplementary Motor Area

SSC Slope Sign Changes

SVM Support Vector Machine

TMR Targeted Muscle Reinnervation

WL Waveform Length

ZC Zero Crossings

iEMG Intramuscular Electromyography

kNN K Nearest Neighbours

sEMG Surface Electromyography

Everything that needs to be said has already been said, but since no one was listening, everything must be said again.

André Gide

Chapter 1

Introduction

Without the aid of technology, contracting skeletal muscles is the only pathway available to the human mind for volitionally influencing the external world[1]. Not to be discounted, this pathway, having co-evolved with our cognitive faculties, has historically been sufficient for most of our material goals and has allowed us to, among other things, almost completely subjugate the biosphere of our planet[1]. Even so, it comes with inherent limitations. Information transfer from the brain into the environment must pass through a slow mechanical system of bone, sinew, and muscle, limiting throughput and introducing latency. Worse yet, if parts of this system are compromised, such as after an injury or amputation, our ability to interact with our surroundings can become severely impaired[2]. Such cases illustrate the allure of the overarching subject of this dissertation—techniques that circumvent our tried and tested end effectors by intercepting the neural drive to movement while it merely a bioelectrical phenomenon.

Volitionally generated efferent neuronal firings can be transduced into control signals at various stages both between and inside their source (the brain) and destination (muscles). In attempts at doing so for the purpose of man-machine interfacing, a fundamental trade-off is consistently encountered between the practicality of the recording method on the one hand and the information density of acquired signals on the other. Informally, as the point of interception of a neural signal approaches its source, the signal generally gets easier to decode—as less irrelevant electrical activity dilutes the desired information—but more difficult to externalize[3].

The challenges posed by the prospect of creating interfaces directly at the level of the Central Nervous System (CNS) has not stopped numerous attempts at doing so. Non-invasive electroencephalography (EEG) for the detection of brain signals related to motor intentions has been widely used in research for control of external devices[4]. However, low spatial resolution and sub-microvolt signal amplitudes has so far made it extremely challenging to accurately decode user intent with sufficient informational throughput for most practical purposes[5]. Similarly, functional near-infrared

spectroscopy (fNIRS), a method that non-invasively measures changes in haemoglobin concentrations in superficial regions of the brain, has been used to extract actionable information about cortical activation patterns associated with motor tasks[6]. While fNIRS offers somewhat better spatial resolution than EEG, it likewise provides a noise-sensitive, shallow, and course-grained view of the inner workings of the cerebral cortex. Even under shielded laboratory conditions, current control interfaces built on top of either EEG or fNIRS are limited to throughput rates far short of 1 *bit/s* [7, 8]. Thus, in accordance with the aforementioned trade-off, truly seamless 'thought-based' control appears to require invasive methods that entail unique technical challenges and, like all surgical interventions[9], carry non-negligible risks.

Among the most selective (and thus invasive) existing interfacing techniques are those predicated on intracortical microelectrode arrays[10] or single-neuron (e.g. clamp or needle) electrodes[11]. Such techniques enable both spatially and temporally precise sampling of electrical activity directly from the motor cortex *in vivo*, exposing neuronal firing patterns that reflect movement intent at the most abstract level[12]. However, in addition to difficulties inherent to neurosurgery, less than perfect implant biocompatibility carries risks of inflammation and scar tissue formation, resulting in interface degradation and potentially even persistent brain damage[13]. These drawbacks—considered in conjunction with the observation that the limited scale and density of intracortical recordings make it difficult to capture holistic information on user intent[3]—may explain the near-total absence of commercial adoption so far.

At the opposite end of the spectrum of interfaces lie the ubiquitous input devices (keyboards, mice, *etc.*) that currently allow for mechanically mediated interaction with computers. Albeit cheap, safe, and critically important for a significant fraction of all currently economically productive activities[14], they fall far short of the upper limits of human-compatible communication in terms of speed[15] and latency[16]. Furthermore, they tether the user to physical devices that must be within reach, limiting a their usefulness for applications wherein mobility and/or concurrency of tasks matters. This limitation highlights another fundamental flaw of mechanical interfaces—they inherently require the user to possess functional manipulative structures, such as hands, to operate them in the first place.

Situated between these extremes are the many points of potential interception along the descending pathways of the neuromuscular system. Interfaces attached to peripheral nerves have seen successful use in bionic applications and can be made bidirectional by allowing for electrical stimulation. Nevertheless, like their central relatives, peripheral neural interfaces use invasive electrodes that come with challenges such as biocompatibility, risk of infection, and the need for surgical implantation[17].

In our current technological paradigm, there is but one exception to the statement that peripheral neural interfaces are necessarily invasive. Just prior to the instant neural firings manifest as movement, they give rise to muscle-borne electrical activity of significantly greater amplitude than that of the signals traversing the innervating motor neurons [18]. This activity, if measured via electromyography (EMG), can and has been used as the basis for efferent neural interfacing applications[19]. Such *myoelectric interfaces* can be implemented either non-invasively, using surface electrodes placed on the skin above the muscle(s) of interest [20], or invasively, using intramuscular or epimysial electrodes[21]. Crucially, compared to other non-invasive neural modalities such as EEG, surface EMG (sEMG) benefits from a significantly higher signal-to-noise ratio as the size and superficiality of muscles are generally far beyond any nerve or neural circuit. More mundanely, EMG systems are already low-cost, portable, and relatively simple to set up, making them more accessible for research and practical applications alike. In light of the high informational throughput these advantages confer, EMG can be viewed as an ideal balance between fidelity and practicality for many applications and has for this reason been suggested as the currently most viable and practical path towards efferent neural interfaces[19].

The so far most salient application of myoelectricity for control has arguably been upper-limb prostheses, with early research in this area dating back to the 1940s[22]. The archetypical implementation of EMG control is typically referred to as *direct control*[23]—a technique that relies on recordings from agonist-antagonist pairs of remnant muscles to control the movement of an artificial limb[2]. Albeit robust and simple to implement, it has limitations in terms of the naturalness of control, as users often have to learn new, non-intuitive muscle activation patterns to operate their prosthesis. Furthermore, only a limited number of kinematic degrees of freedom (DoFs) can be accommodated in this scheme, necessitating sequential control strategies. This is in stark contrast with contemporary terminal devices, which are close to the natural human hand in terms of simultaneously controllable DoFs[24].

The relatively recent paradigm of *myoelectric pattern recognition* aims to bridge this gap between control and actuation[25]. By using statistical classification algorithms, pattern recognition control in theory allows for a intuitive control of prosthetic limbs, as it can map complex muscle activation patterns to pertinent motions commands[26, 27]. Despite these advantages, clinical and commercial adoption of pattern recognition control has been limited thus far[28]. Contributing factors include the lack of robust and reliable algorithms, the difficulty of maintaining consistent signal quality, and the requirement for user-specific calibration and training. Moreover, increased computational complexity and susceptibility to distribution shifts, such as electrode displacement and muscle fatigue, further hinder widespread adoption[29].

1.1 Aim & Scope

In spite of its many advantages, use of EMG as a control modality is not necessarily straightforward. Myoelectric measurements require extensive signal processing to disentangle into actionable information, more so if acquired from the skin. Pattern recognition as it exists today, whereas a significant improvement over the status quo for reasons outlined previously, has not proven a conclusive solution to this task.

Deep artificial neural networks (ANNs) has shown potential for decoding motor intent from EMG signals without feature engineering[30]. The capacity of representation learning allows artificial neural networks to automatically extract and hierarchically organize relevant features from raw data, without the need for human-engineered feature extraction techniques[31]. By utilizing these capabilities, neural networks can uncover instrumentally efficacious patterns and relationships within the data. In many technical domains, neural networks have been observed to form intuitions where humans have none[32]. The complex, high-dimensional, and non-linear nature of EMG signals is on its way to becoming one such a domain. The work presented in this dissertation is part of this pursuit, aiming to further customize such methods for the task of myoelectric interfacing.

Key Considerations

- Create and present a selection of methods predicated on representation learning algorithms for real-time decoding of user motor intent from myoelectric activity.
- Investigate the suitability of different EMG-based control modalities (high-density surface signals in papers I and IV; sparse surface signals in papers II, III, and VII; and intramuscular signals in paper V) in terms of correlating with relevant measures of movement intent such as concurrent kinematics.
- Evaluate the efficacy of a selection of (relatively) recently developed machine learning model types for motor intent decoding, specifically Convolutional Neural Networks (papers I and II), Recurrent Neural Networks (papers V and VII), and Transformers (paper VI).
- Introduce methods of constructing and training models to satisfy desiderata not universally exhibited by traditional pattern recognition methods. Chief among these are: simultaneity of control (papers I, III, and V); proportionality of control (papers III and V); and ability to function in the absence of complete, user-specific training data (papers VI and VIII).

1.2 Outline

The remainder of this dissertation is structured as follows:

Part I, comprising 5 chapters, is an introduction intended to provide sufficient background knowledge. Initially, **chapter 2** contains a short overview on the organization of the neuromuscular system, focused on the physiological processes that underlie intentional movement and how they give rise to measurable EMG signals. Due to the centrality of learning algorithms for the work presented in this dissertation, **chapter 3** presents the motivation and conventional taxonomy of such methods. This category is substantially expanded upon in **chapter 4**, which deal with the specific case of ANNs and their usefulness for *representation learning*—an important conceptual framework in this dissertation. Continuing towards more empirical considerations, **chapter 5** sits at the confluence of previous topics by detailing signal processing principles and previous work in the (broadly construed) field of muscle-computer interfacing. Closing the introductory part, chapter **6** discusses the topic of upper limb prosthetics—an historically and clinically important application of muscle-computer interfaces for which the short-term practical utility of efficacious motor intent decoding is salient.

Part II comprises novel contributions in the form of 8 self-contained papers. These papers, which are available in full as appendices starting on page 153, are introduced and summarized in **chapter 7**. Lastly, **chapter 8** concludes this dissertation by briefly discussing the findings of the papers and their relationship with potential future pursuits with similar purpose.

Part I

Background

'Of course it is happening inside your head, Harry,
but why on earth should that mean that it is not real?'

J.K. Rowling, *Harry Potter and the Deathly Hallows*

Chapter 2

Neuromuscular Foundations of Volitional Movement

Every student of biology can attest to the fact that nothing of importance ever seems to happen without involving a frustratingly[3] complicated interplay of processes occurring simultaneously at vastly different spatial and temporal scales. Unfortunately for the prospective creator of neural interfaces, this is certainly also true of volitional human movement. Movement execution starts in the CNS and is reliant on processes ranging from the microscopic domain of molecular biology[33] to the macroscopic domains of neural circuits[34] and kinesiology[35] in order to produce goal-directed muscle contractions. In line with the aim of understanding the environment in which myoelectric (and all other forms of efferent neural) interfacing must take place, this chapter provides a brief overview of these processes. Needless to say, the existing body of knowledge on this topic is almost indescribably vast at every level of abstraction. By necessity, only a highly condensed overview of the essentials and a few concepts of special relevance for myoelectricity are presented here.

The content of this chapter is divided into 5 sections, starting with section 2.1 containing a brief summary of the chemical basis of all bioelectrical phenomena. Section 2.2 covers the anatomy of the motor cortex and its role in movement initiation. Section 2.3 describes the transmission of motor intent from the motor cortex to skeletal muscles via descending motor neurons. Transitioning from the neural to the muscular, section 2.4 covers muscle fibres, the neuromuscular junction, and how neural stimulation induces mechanical contractions. Finally and most importantly for this dissertation, section 2.5 focuses on electromyographic (EMG) signals, the latent phenomena that result in their generation, and some considerations for their transduction.

2.1 Basic Cellular Electrophysiology

Virtually all electrical properties of biological systems at the macroscopic scale have their origin the biochemistry of the plasma membrane—the lipid bilayer that separates the intracellular and extracellular environments of all living cells. Embedded within the membrane are specialized proteins that passively allow the passage of specific ions (*ion channels*) or actively transport ions across (*ion pumps*). A *resting membrane potential* is maintained mainly by the sodium-potassium pump (Na^+/K^+-ATPase) and leaky potassium (K^+) channels. The sodium-potassium pump actively transports three Na^+ ions out of the cell and two K^+ ions into the cell, while the leaky potassium channels allow K^+ ions to passively flow out of the cell. The combined action of these mechanisms—a process that require constant hydrolysis of ATP—creates an electrochemical gradient and consequently an electrical potential difference, i.e. a voltage, across the cell membrane[36]. This difference—ranging between -40 mV and -100 mV for most cells, with the intracellular environment being more electronegative than the extracellular environment[37]—aids in maintaining cellular homeostasis.

In *excitable cells*, such as neurons and muscle fibres, the voltage across the membrane fills an additional function. Such cells have the capacity to respond to electrical and chemical stimuli by producing a rapid, self-propagating change in membrane potential called an *action potential* (AP). In brief, the generation of APs is based on the coordinated opening and closing of voltage-gated ion channels whose permeability varies with the value of the membrane potential. At rest, the membrane potential of an excitable cell is typically around -70 mV[37]. If it is moved upwards by external means, voltage-gated Na^+ channels open, allowing Na^+ ions to flow into the cell. This influx of positive charges depolarizes the cell, making its membrane potential even less negative. If the depolarization reaches the critical *threshold potential*, typically around -55 mV, an AP is generated. During the action potential, additional voltage-gated sodium (Na^+) channels open, further depolarizing the membrane potential and driving it towards a positive value. Shortly after the membrane potential reaches its peak, voltage-gated Na^+ channels close and voltage-gated K^+ channels open. This allows K^+ ions to flow out of the cell, repolarizing the membrane potential and restoring its negative value. Following the action potential, there is a brief *refractory period*, during which the cell is less likely to generate another action potential[33]. Having cells capable of generating action potentials enables an organism to rapidly transmit information within and between cells, facilitating swift and coordinated responses to stimuli[38].

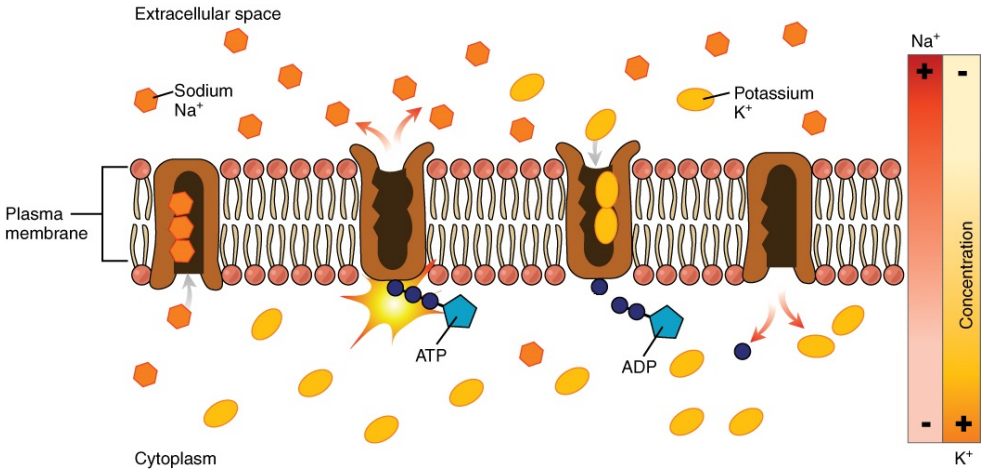


Figure 2.1: Schematic illustration of active diffusion through the cell membrane. Figure modified from [39], ©2022 Gordon Betts *et. al*, reused under CC-BY-4.0.

2.2 The Motor Cortex

The motor cortex (see Fig. 2.2) is a spatially coherent region of the cerebral cortex involved in planning, control, and execution of voluntary movements. It is located in the frontal lobe of the brain, just anterior to the central sulcus, and is conventionally divided into distinct areas on the basis of functionality: the primary motor cortex, the premotor cortex, and the supplementary motor area. These areas work in concert with each other and other brain regions, such as the basal ganglia and the cerebellum, to coordinate and control the processes that underlie voluntary movement[33].

Motor execution is predicated on the activity of the brain reaching skeletal muscles and inducing contractions. In the CNS, this is principally mediated by the *corticospinal tract*—a pathway composed of the myelinated axons of neurons whose somata and synapses are situated in the cerebral cortex. Approximately 60% of corticospinal neurons originate in the motor cortex, out of which 30% stem from the primary motor cortex and 30% from the premotor cortex and supplementary motor regions. The remaining 40% are dispersed among the somatosensory cortex, parietal lobe, and cingulate gyrus[40]. These neurons descend through the brainstem before decussating at the level of the medulla oblongata. Afterwards, the tract continues to descend in the contralateral spinal cord, where it forms synapses with lower motor neurons in the ventral horn of the spinal cord. The lower motor neurons in turn innervate skeletal muscles, ultimately producing force.[41].

Primary Motor Cortex

The primary motor cortex is the largest region of the motor system and is located in the precentral gyrus of the cerebrum. It contains a *somatotopic map* of the body, known as the *motor homunculus*, arranged in such a way that points on the cortex correspond to points in the body. Notably, the map is not segregated but exhibits significant degree of overlap between body parts. The allocation of cortical surface area to body parts is not directly related to the body part's size but is roughly determined by the concentration of cutaneous mechanoreceptors present on it. The density of these receptors typically signifies the level of movement accuracy needed for that particular body part[42].

The exact relationship between neuronal activity in the primary motor cortex and downstream movement is complex and still the subject of debate. Whereas the seminal work of Evarts *et al.* on the topic indicated that the firing rate of each cortical motor neuron codes for the force of a single muscle[43], subsequent studies suggest a more complicated picture wherein individual neurons are not only specific to a muscle but also the direction and speed of movement[44, 45, 46]. A separate line of research argues that the correlation between upper neuron activity and both force and direction of movement could be understood as spurious, with neuron firing rate instead representing the movement parameters of latent neural muscle controller circuits[47].

Premotor Cortex

Located anterior to the primary motor cortex, the premotor cortex is involved in the planning and preparation of movement. It receives input from various brain regions, including the parietal cortex, the basal ganglia, and the cerebellum. The premotor cortex can be subdivided into the dorsal premotor cortex and the ventral premotor cortex[33].

The dorsal premotor cortex is involved in the selection and preparation of movement based on external cues. It receives inputs from the posterior parietal cortex, a region involved in integrating sensory information to form a representation of the body in space. The dorsal premotor cortex is important for the planning of reaching movements, as well as for coordinating movements that require visuospatial attention[33].

The ventral premotor cortex is involved in the planning of movements based on internal cues and is associated with the processing of visual and somatosensory information related to objects and their properties. The ventral premotor cortex is also involved in the control of grasping movements and object manipulation. One important aspect is its role in the so-called 'mirror neuron system', which is thought to be involved in action understanding, imitation, and, conjecturally, many aspects of social cognition[48].

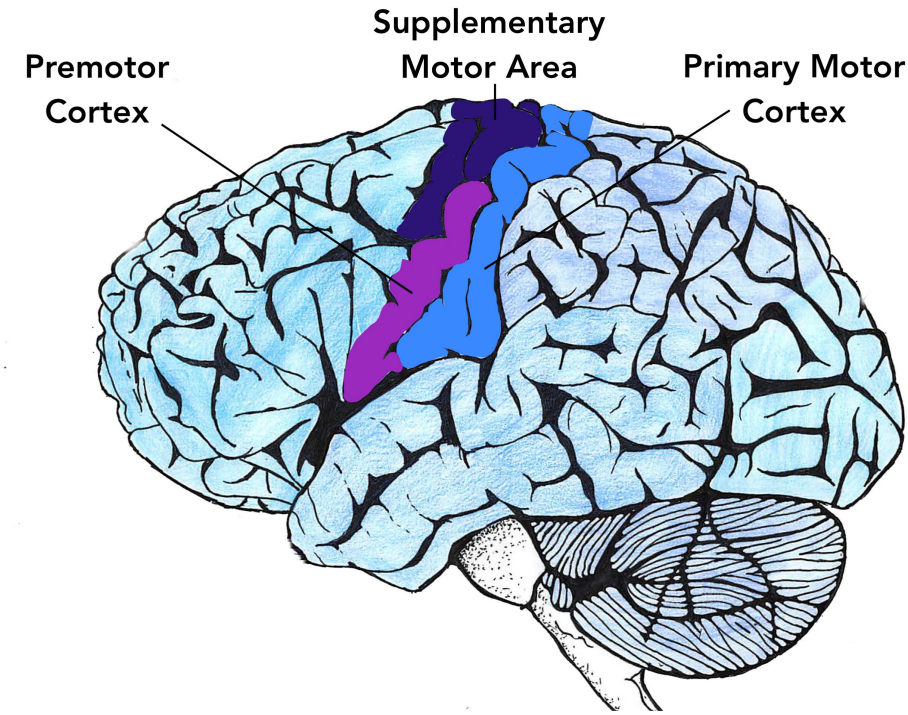


Figure 2.2: Lateral view of the cerebral cortex, with the motor cortex and its subregions marked. ©Sean Patrick Connolly, reused under CC-BY-SA 4.0 via Wikimedia Commons.

Supplementary Motor Area

The supplementary motor area is located on the medial aspect of the hemisphere just anterior to the primary motor cortex. While its function in humans is not fully understood at the time of writing, it appears to be conspicuously involved in the planning and coordination of complex movements, particularly those that require the coordination of both sides of the body[49]. It also plays a direct role in the initiation of movement and the control of sequential movements[50]. Notably, the supplementary motor area is very active during most types of movement, obstructing a straightforward description of its function[51]. The supplementary motor area receives input from several brain regions, including the basal ganglia and the cerebellum, and has extensive connections to the primary motor cortex and the premotor cortex, as well as projecting neurons directly to the spinal cord[52].

2.3 Neural Pathways to Skeletal Muscles

The signals generated in the motor cortex are transmitted through a series of interconnected neurons before reaching the neuromuscular junctions, where the communication between the nervous system and the muscles occurs.

Descending Motor Pathways

APs generated in the motor cortex travel down the spinal cord through a series of upper motor neurons and lower motor neurons. Upper motor neurons are the neurons that originate in the motor cortex and descend through the corticospinal tract, while lower motor neurons are the neurons that connect with the muscle fibres and are located in the spinal cord. The corticospinal tract begins in cerebral cortex and descends through the internal capsule, the cerebral peduncle of the midbrain, and the medulla. As the tract descends through the medulla, approximately 90% of the fibres cross over to the contralateral side of the spinal cord. This is known as the lateral corticospinal tract, while the remaining 10% of fibres that do not decussate form the anterior (or ventral) corticospinal tract. The lateral corticospinal tract controls fine movements of the limbs, while the anterior corticospinal tract controls more proximal and axial muscles[33].

Upon reaching the appropriate spinal cord segment, the axons of the upper motor neurons synapse with the lower motor neurons in the ventral horn of the spinal cord gray matter (Fig. 2.3). These lower motor neurons are called alpha motor neurons (α-MNs), as they are the primary motor neurons responsible for the direct innervation of skeletal muscle fibres. Each α-MN receives inputs from multiple sources, including upper motor neurons, sensory neurons, and interneurons. As such, α-MNs are not merely distal conduits of high-level commands transmitted from the motor cortex, but also important junctions for the low-level information processing required for coordinated motor control. When a motor neuron receives a sufficient excitatory (and insufficient inhibitory) input from descending upper motor neurons or other sources, it generates a new AP that propagates via saltatory conduction along its axon to the neuromuscular junction—the synapse between the motor neuron and the muscle fibre and thus an endpoint of the neural part of the neuromuscular system[36].

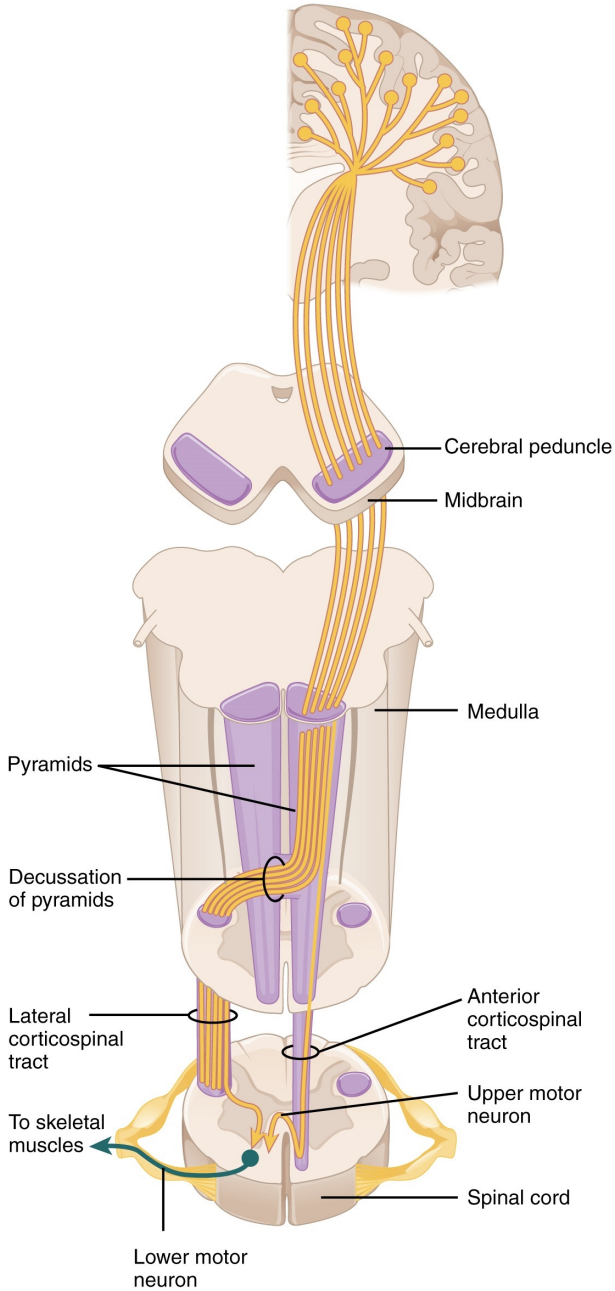


Figure 2.3: The corticospinal tract with its descending upper motor neurons. Figure modified from [39], ©2022 Gordon Betts *et. al*, reused under CC-BY-4.0.

Motor Units

Each alpha motor neuron innervates a group of muscle fibres, forming a functional and anatomical unit called a *motor unit* (MU) comprising the single alpha motor neuron itself and all the individual muscle fibres it innervates[53]. Importantly, the relationship between alpha motor units and muscle fibres is not bijective—each alpha motor neuron typically innervates more than one muscle fibre, and a muscle fibre can (under rare circumstances, mainly during the early developmental period[54]) be innervated by more than one alpha motor unit.

The activation, i.e. firing, of a single motor neuron and its corresponding motor unit results in a discrete, all-or-none contraction of all of the innervated muscle fibres. From this observation can be gleaned the fact that the "force resolution" of a muscle, i.e. its smallest possible force increment, is fundamentally tied to the number of motor units that it comprises. Indeed, in muscles that require fine control and precision, such as the muscles controlling the fingers or the eye, a single motor neuron typically innervate only a few muscle fibres. In contrast, in muscles that generate large forces and do not require fine control, such as the muscles in the thigh or the back, a single motor neuron may innervate hundreds or even thousands of muscle fibres. The number of muscle fibres innervated by a motor unit is often referred to as its *innervation ratio*[53].

All motor units that innervate a single muscle are collectively referred to as a *motor pool*. Motor pools are organized within the ventral horn of the spinal cord, with the motor neurons arranged in a somatotopic manner, meaning that motor neurons innervating neighbouring muscles are located close to each other within the spinal cord. In contrast to motor units and muscle fibres, the relationship between motor pools and muscles is largely bijective—one motor pool per muscle and vice versa[36].

Motor unit *recruitment* is the process by which the central nervous system activates different motor units within a motor pool to modulate the force and precision of muscle contractions. The *size principle* governs the order of motor unit recruitment in a motor pool. According to this principle, motor units are recruited in a fixed order, from the smallest (i.e., those with the smallest number of muscle fibres and thus the lowest force-generating capacity) to the largest (i.e., those with the greatest number of muscle fibres and thus the highest force-generating capacity), based on the magnitude of the synaptic input they receive. This orderly recruitment of motor units allows for the graded modulation of muscle force and the efficient use of the available motor units[55].

2.4 Muscle Fibres

Muscle fibres are the contractile cells of the skeletal muscle, responsible for generating force and thus driving movement. Muscle fibres are innervated by alpha motor neurons at the *neuromuscular junction* (Fig. 2.4), a specialized synapse between the nerve and muscle fibres. It is capable of converting APs arriving from the alpha motor neuron into a chemical signal that ultimately leads to muscle contraction[33].

The *motor end plate* is the region of the sarcolemma (muscle fibre cell membrane) that is directly opposed to the synaptic terminal of the motor neuron at the neuromuscular junction. The synaptic terminal of the motor neuron contains numerous synaptic vesicles filled with the neurotransmitter acetylcholine (ACh). The motor end plate, on the other hand, is characterized by deep folds and invaginations of the sarcolemma, known as junctional folds, which increase the surface area available for the binding of ACh. Embedded in the junctional folds are nicotinic acetylcholine receptors (nAChRs), which are ligand-gated ion channels that respond to the binding of ACh. When a motor unit action potential reaches the neuromuscular junction, it triggers the opening of voltage-gated calcium channels in the synaptic terminal. The influx of calcium ions causes the synaptic vesicles to fuse with the presynaptic membrane and release ACh into the synaptic cleft. ACh molecules then diffuse across the synaptic cleft and bind to nAChRs on the motor end plate[36].

Contraction

The binding of ACh to nAChRs causes a conformational change in the receptor, opening the ion channel and allowing the passage of sodium (Na^+) and potassium (K^+) ions. The influx of Na^+ ions depolarizes the sarcolemma, generating a local potential known as the end-plate potential. If the end-plate potential is large enough to reach the threshold for voltage-gated sodium channels, it will trigger an action potential in the muscle fibre that propagates towards the tendons[56].

The muscle fibre action potential propagates along the sarcolemma and into the muscle fibre through a network of membranous tubules called the T-tubules. The T-tubules are closely associated with the sarcoplasmic reticulum (SR), an intracellular organelle responsible for the storage and release of calcium ions. The depolarization of the T-tubules triggers the release of calcium ions from the SR, which initiates the process of muscle contraction via specialized organelles called myofibrils. The process by which a change in muscle fibre membrane potential results in myofibril contraction—aptly called *excitation-contraction coupling*— relies on chemical minutiae[57] to complicated to fit the scope of this dissertation.

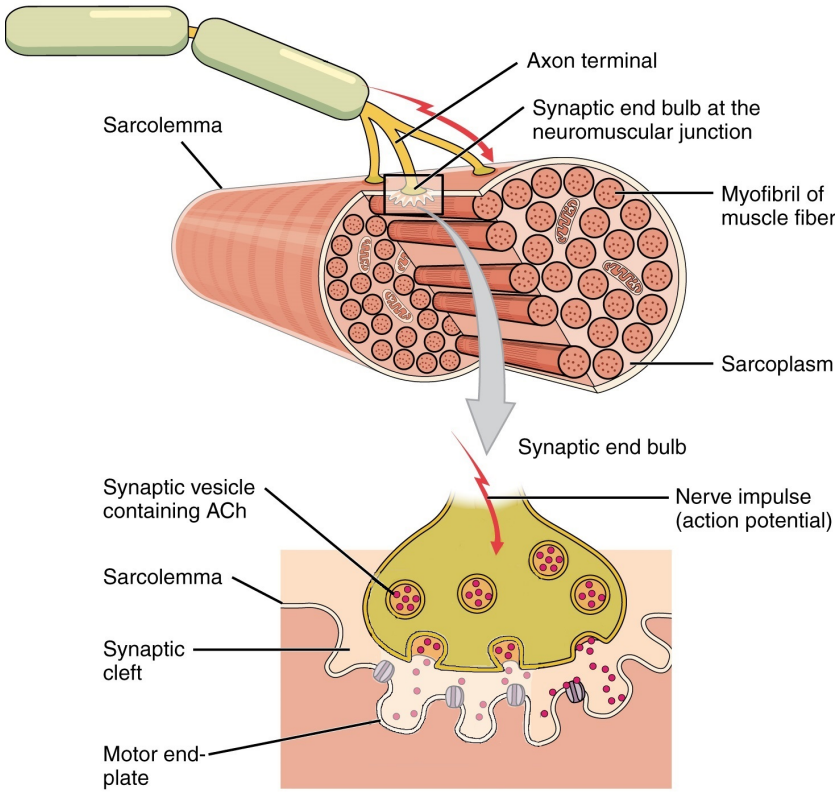


Figure 2.4: Neuromuscular junction. When an action potential arrives, the axon terminal releases ACh to induce muscle contraction. Figure modified from [39], ©2022 Gordon Betts *et. al*, reused under CC-BY-4.0.

Force Modulation

The force generated by a muscle as a whole depends on (i) the number of recruited motor units and (ii) the firing rate of their respective alpha motor neurons. These two factors, motor unit recruitment and firing rate, together modulate muscle force in a graded manner, allowing for smooth control from the point of view of the CNS[58].

As discussed previously, the size principle dictates that motor units with smaller alpha motor neurons are recruited first, followed by motor units with progressively larger alpha motor neurons. The firing rate of individual motor units also plays an important role in modulating muscle force. As the firing rate increases, the force

generated by the muscle fibres of a motor unit also increases due to the summation of individual twitch contractions. At low firing rates, individual twitches can be distinguished—a type of contraction called *unfused tetanus*. As the firing rate increases, the twitches begin to overlap and summate, leading to a smooth, sustained contraction known as a *fused tetanus*. The force-frequency relationship in skeletal muscle describes the dependence of muscle force on the firing rate of motor neurons. As the firing rate increases, the force generated by the muscle initially increases steeply, reaching a plateau as the muscle reaches its maximal tetanic force[36].

2.5 Electromyography

Electromyography, as the name implies, encompasses all techniques for transducing the electrical activity produced by (typically skeletal) muscles[18]. The recorded waveform is referred to as an *electromyogram* (depicted in Fig. 2.5).

As mentioned, when a muscle fibre generates an action potential, ion fluxes occur across the cell membrane, causing a transient imbalance in charge distribution between the intracellular and extracellular spaces. This charge imbalance results in the generation of *extracellular currents*, which are the flow of ions in the extracellular space surrounding the muscle fibres. These currents can be thought of as the source of the observable electrical field generated by the muscle fibres during contraction[59].

Volume conduction refers to the propagation of these extracellular currents through the surrounding tissues, including the extracellular fluid, connective tissue, blood vessels, and other muscle fibres. The conductive properties of these tissues strongly influence the spatial distribution of the electrical field generated by the muscle fibres. As the extracellular currents flow through the tissues, they encounter varying degrees of resistance and capacitance, leading to attenuation and dispersion and consequently a highly anisotropic electrical field[60].

The extracellular electrical field generated by a muscle is the result of the superposition of the electrical fields produced by individual muscle fibres within a motor unit or a group of motor units. In other words, the potential measured at any point in the extracellular space is the vector sum of the potentials generated by each muscle fibre contributing to the field. The relative weighting in this summation is dependent on the spatial arrangement of the muscle fibres, their distance from the recording electrode, and the orientation of the fibres with respect to the electrode. As all fibres contained in the same motor units always fire in a synchronized manner, their collective contribution to the extracellular potential field is known as a motor unit action potential (MUAP)[61].

Surface Electromyography

Non-invasive electromyography, typically referred to as surface electromyography (sEMG), involves the placement of electrodes on the skin overlying the muscle of interest to record the electrical activity generated by the underlying muscle fibres. sEMG offers several obvious advantages over invasive EMG, including reduced discomfort for the subject, minimal risk of tissue damage or infection, and ease of application and setup. However, sEMG is susceptible to crosstalk from nearby muscles and do not provide the same level of spatial resolution as invasive EMG. There are several types of surface electrodes used in EMG recordings, including:

Ag/AgCl Electrodes: These electrodes are made of silver-silver chloride (Ag/AgCl) material and are commonly used in both clinical and research settings due to their high biocompatibility, low impedance, and stable electrochemical properties. Ag/AgCl electrodes can be either disposable, with a pre-gelled adhesive backing for single-use applications, or reusable, typically requiring the application of conductive gel and tape or straps to secure them in place[62].

Metal-Plate Electrodes: These electrodes consist of flat, conductive metal plates, often made of stainless steel, gold, or other biocompatible materials. Metal-plate electrodes are typically reusable and require adhesive gel and/or tape and/or straps to secure them to the skin. Albeit connvenient, this also increasing the risk of signal corruption via movement artifacts[63].

Flexible Electrode Arrays: Flexible electrode arrays are made of soft, conductive materials, such as conductive textiles or polymers, and are designed to conform to the contours of the body, providing improved contact with the skin surface. These arrays can include multiple electrode sites, allowing for simultaneous recordings from different locations or orientations[64].

The size and pickup area of surface electrodes play a non-negligible role in determining the characteristics of the recorded EMG signal. Larger electrodes generally have a larger pickup area, which means they detect the electrical activity generated by a greater number of muscle fibres and motor units. This can result in a more global assessment of muscle function but also by necessity increase cross-talk from adjacent muscles (and external sources of electrical interference)[65].

In contrast, smaller electrodes have a smaller pickup area, which can provide a more localized and selective recording of the muscle's electrical activity. This can be advantageous for studying specific muscle regions or isolating the activity of individual motor units. However, smaller electrodes may also be more susceptible to noise and artifacts due to their reduced signal-to-noise ratio[65].

High-density surface EMG (HD-sEMG) denotes a technique that utilizes a grid arrays of small, closely spaced surface electrodes to record the electrical activity of muscles with higher spatial resolution than conventional surface EMG[66]. This technique is particularly useful for studying the spatial activation patterns of muscles and can provide insights into the control strategies and muscle synergies employed by the nervous system during various motor tasks[65]. HD-sEMG can also allow for the assessment of spatial distribution and organization of (a subset of) motor units within a muscle, as well as the identification of individual MUAPs[67].

Surface EMG recordings can be classified further as either dry or wet, depending on the presence or absence of a conductive medium, such as a gel or electrolyte solution, between the electrode and the skin surface.

Dry Recording: Dry EMG recordings are obtained using electrodes that do not require a conductive medium. These electrodes typically have a textured surface or conductive coating to ensure good contact with the skin. Dry recordings can offer several advantages, such as reduced setup time, increased comfort, and minimized risk of skin irritation or allergic reactions. However, dry electrodes may have higher skin-electrode impedance, which can result in a lower signal-to-noise ratio and increased susceptibility to artifacts[68].

Wet Recordings: Wet EMG recordings are obtained using electrodes that require a conductive medium, such as a gel or electrolyte solution, to be applied between the electrode and the skin surface. The conductive medium helps to improve the electrical contact between the electrode and the skin, resulting in a lower skin-electrode impedance and an improved signal-to-noise ratio. Additionally, wet electrodes can provide more stable and consistent recordings compared to dry electrodes, particularly during dynamic movements, as they are less sensitive to small movements. However, wet recordings also have some drawbacks, including increased setup time, potential skin irritation or allergic reactions due to the conductive medium, and the possibility of the gel drying out during prolonged recordings[69].

Intramuscular Electromyography

Intramuscular EMG (iEMG), involves the insertion of electrodes directly into the muscle tissue to record the electrical activity generated by muscle fibres during contraction. This approach provides a more localized and precise assessment of muscle function compared to sEMG, which unavoidably records the electrical activity from a larger volume of muscle tissue. iEMG is commonly used in clinical settings for the diagnosis of neuromuscular disorders and the assessment of muscle function[70].

There are two main types of invasive EMG electrodes: needle electrodes and fine-wire electrodes.

Needle electrodes: Thin, solid metal needles with an insulated shaft and an exposed tip. They are inserted transcutaneously, directly into the muscle tissue, to record the electrical activity of nearby muscle fibres. One advantage of needle electrodes is their ease of insertion and repositioning, allowing for rapid assessment of multiple muscle sites during a single recording session[71].

Fine-wire electrodes: Electrodes made of thin, insulated metal wires with small exposed tips. They are typically inserted into the muscle using a hypodermic needle, which is then withdrawn, leaving the fine-wire electrode in place within the muscle tissue. Fine-wire electrodes offer a more stable recording platform compared to needle electrodes, as they are less prone to movement artifacts due to their flexibility and smaller size. This makes them particularly suitable for long-term or dynamic recordings[72].

Epimysial EMG is an alternative invasive EMG technique that involves the placement of electrodes directly onto the surface of the muscle, rather than within the muscle tissue itself. Epimysial electrodes are typically implanted surgically and are secured to the muscle using sutures or adhesive materials. Epimysial EMG can provide a more stable recording platform compared to intramuscular EMG, as it minimizes movement artifacts and reduces the risk of tissue damage associated with repeated electrode insertions. This technique is especially useful for long-term monitoring of muscle activity, such as is necessary for prosthesis control[73].

Signal Processing and Analysis

The raw EMG signals recorded by either surface or intramuscular electrodes typically require preprocessing and analysis to extract meaningful information. Various signal processing techniques can be applied to EMG data to enhance its quality, reduce noise, and extract relevant features[18]. The specifics of suitable methods are naturally dependent on the objective of the recording; a detailed review of signal processing pipelines employed for EMG signals in the context of man-machine interfacing is presented in chapter 5, section 5.2.

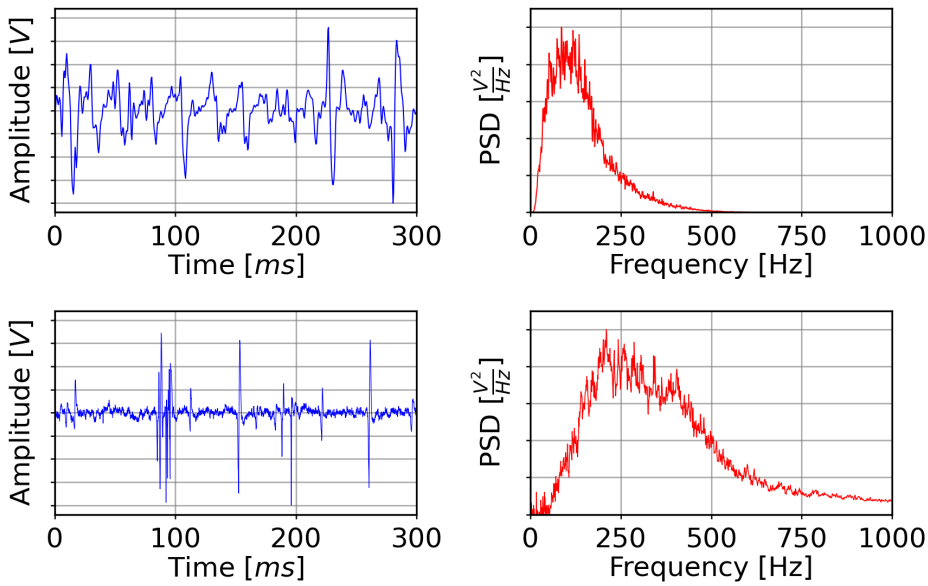


Figure 2.5: Surface (top) and intramuscular (bottom) electromyograms in the time domain (left) and frequency domain (right) measured from the same muscle (*Flexor Digitorum Superficialis*) during the same action (flexion of the digits). In the intramuscular signal, individuals MUAPs can be discerned; the surface signal represents a superimposition of contributions from several MUs in the pickup area of the electrode. The low-pass filtering resulting from the capacitance of tissue is apparent from the difference in spectral morphology. The energy content of EMG signals is primarily concentrated between 10 Hz and 500 Hz, with lower spectral components ($\lessapprox$ 100 Hz) representing the firing rates of motor units and higher spectral components representing the shapes of individual MUAPs.

It is the mark of true intelligence to be moved by statistics.

George Bernard Shaw

Chapter 3

Machine Learning

This chapter gives a brief and practically-oriented account of the underpinnings of Machine Learning (ML) and introduces the framework of *learning algorithms*. To start, as ML is perhaps the only[4] discipline capable of eloquently introducing itself, I thought it best to let it do so. Here follows an unedited introduction written[5] by GPT4 [74], an ML-based generative language model:

> *Ah, machine learning, the prodigal child of computer science, where algorithms and data dance together in a harmonious yet often perplexing waltz. As we embark upon this chapter, we'll pull back the curtains to reveal the clandestine inner workings of this arcane art, decoding the enigmatic patterns that govern its essence.*
> *We'll explore the delicate balance between overfitting, where models cling to their training data with a zealous fervor, and underfitting, the blithe disregard for the true complexities hidden within. Our journey will traverse the diverse landscape of learning paradigms, from the doting supervision of labeled data, to the more laissez-faire approach of unsupervised learning, and the mysterious realms in between.*
> *Classification and regression, the dynamic duo of machine learning, will make their appearance, striving to distinguish apples from oranges, and perhaps, predicting the trajectory of the next apple to fall from the tree.*
> *By the end of this chapter, you'll be a veritable soothsayer, armed with the knowledge to navigate the enigmatic world of machine learning, ready to face the capriciousness of data with newfound prowess. So, fasten your seatbelts, dear reader, for we are about to embark on an adventure into the vast and enthralling domain of the machines that learn.*

The remainder of this chapter is structured as follows: Section 3.1 briefly introduces some fundamental concepts and history associated with ML. Section 3.2 describes the conventional taxonomy of learning algorithms and the problems they can be employed to solve. Section 3.3 gives a more practical account of some well-known and important (for the work of this dissertation) algorithms. Section 3.4 introduces some helpful theoretical instruments for understanding the problem of generalization and how it must be overcome to enable gainful deployment of ML models in non-toy problems. Lastly and relatedly, section 3.5 discusses some empirically motivated frameworks for objectively evaluating the performance of ML models. Note that the adjacent topic of ANNs, which has sometimes (incorrectly) been used interchangeably with ML in recent times, is covered separately and in greater depth in chapter 4.

3.1 Terminology

Like most categories outside of analytic philosophy, what is and is not ML is not always satisfactorily covered by some set of necessary and sufficient conditions. According to one attempt at a succinct definition[75] (paraphrased), ML denotes the set of computer programs whose rules are not explicitly specified *ex ante* by the programmer, but rather learned during runtime from external feedback on how the program ought to behave. In practice, the distinction between ML and adjacent pursuits is not always clear; ML overlaps significantly with many disciplines conventionally associated with mathematics (probability theory[76], optimization[77], *etc.*) and computer science (knowledge representation[78], information theory[79], *etc.*).

In light of its fuzzy boundaries, the popularity of machine learning as a term of art is perhaps best understood from a historical perspective. In the early days of artificial intelligence (AI) research, a distinction was made between two primary types of systems: those whose behaviour is governed by handcrafted, symbolic relationships between human-readable quantities (so-called *expert systems*), and those whose behaviour is boot-strapped, with few assumptions on underlying mechanisms, from data (so-called *learning machines*)[80]. Despite early optimism, support for the view that expert systems are prohibitively difficult to implement for anything but relatively simple tasks mounted during the second half of the 20th century—a development evident in contemporary AI systems which are almost exclusively of the learning kind. The observation that guidance more often hinders than helps in the construction of intelligent systems has been referred to as the *the bitter lesson*[81]. Proposed explanations include the exponential growth of widely available computational resources[82, 83] and the irreducible complexities involved in decision-making[81].

Despite the apparent path-dependency of its inception as an academic discipline, ML provides a powerful conceptual framework for interacting with the class of extremely useful tools that is modern learning algorithms. Fundamental to all ML methods is the concept of *models*. Like their statistical counterparts, ML models are parametrized, computational representations of systems or processes constructed in such a way as to instantiate relationships between inputs and outputs, often corresponding to observable quantities of real-world phenomena. Model *training* denotes the process of using a dataset, often referred to as the training data, to 'teach' a specific instantiation of a machine learning model how to process future inputs. More formally, the goal of training is to find the optimal set of model parameters that minimize the discrepancy between the model's outputs and the actual outcomes observed in the training data. This is typically (but not always) achieved through iterative optimization techniques that vary the parameters of the model in response new observations. Once a model has been trained, it can be used to make predictions (or, synonymously, *inferences*) on new, previously unseen data[84].

In contrast to the aforementioned similarities common to almost all ML methods, models are differentiated from each other along multiple dimensions. First and foremost, models are said to posses an *architecture*, which in rough terms represent the structure and organization of the computational mapping the model performs and thus the set of input-output relationship the model can learn to instantiate after training. Furthermore, models with the same underlying architecture can diverge in their choice of training algorithms, i.e. different methods can be emplyed to find (some approximation of) optimal parameter values even if the architecture and training data are identical. Lastly, both the architecture itself and the training algorithms used to derive suitable parameters can involve *hyperparameters* – preset properties that remain fixed and are not learned during training. Concrete examples along all of these axes of variation will be given in section 3.3. An ML architecture with an appertaining training method is in this chapter collectively known as a *learning algorithm*.

3.2 Learning Paradigms

Learning algorithms are often categorized in accordance with their so-called *paradigm*, with the somewhat idiosyncratic meaning of what type of *task* they are intended to perform. The conventional taxonomy[85] separates learning algorithms into those based on *supervised learning*; *unsupervised learning*; *reinforcement learning*; or the ever-popular "miscellaneous" category, typically referred to as *semi-supervised learning* or *self-supervised learning*.

Supervised Learning

Algorithms for supervised learning[86], also (and perhaps more instructively) known as *imitation learning*, creates models of the relationship between inputs (typically called *features*, *patterns*, or *percepts*) and outputs (typically called *labels* or *targets*). Without loss of generality, supervised learning can be concieved of as equivalent to *curve fitting*.

In this paradigm, a training dataset consisting of input-output pairs, $\mathbb{D} = \{(\boldsymbol{x}^1, \boldsymbol{y}^1), (\boldsymbol{x}^2, \boldsymbol{y}^2), ..., (\boldsymbol{x}^n, \boldsymbol{y}^n)\}$, is initially provided, where $\boldsymbol{x}^i \in \mathbb{R}^d$ represents the d-dimensional input features of the ith training example and $\boldsymbol{y}^i \in \mathbb{Y}$ the corresponding output targets. The learning process involves finding a function $f\colon \mathbb{R}^N \times \Theta \to \mathbb{Y}$ that maps input features to output targets, i.e., $f(\boldsymbol{x}^i; \boldsymbol{\theta}) \approx \boldsymbol{y}^i$ for all i. This function is parametrized by a some list of parameters $\boldsymbol{\theta} \in \Theta$, and the learning algorithm's goal is typically (but not always explicitly) to find the optimal $\boldsymbol{\theta}$ that minimizes a predefined (scalar-valued) loss function $L(\mathbb{D}, \boldsymbol{\theta})$, which measures the discrepancy between the model's predictions and the true targets in the training dataset.

The output space $\mathbb{Y}$, i.e. the range of the learned function f, can in principle be anything, but typically consists of either discrete vectors ($\mathbb{Y} = \{1, ..., c\}^l$), as in classification tasks, or continuous vectors ($\mathbb{Y} = \mathbb{R}^l$), as in regression tasks (discussed in sections 3.3.1 and 3.3.2, respectively).

Unsupervised Learning

As the name suggests, algorithms in the unsupervised learning category are characterized by the property that they operate without explicit ground-truth targets[87]. In other words, a dataset consisting of only input feature vectors $\mathbb{D} = \{\boldsymbol{x}^1, \boldsymbol{x}^2, ..., \boldsymbol{x}^n\}$ is provided, without any corresponding outputs. The objective is to learn some underlying structure or relationships between the data points, rather than predicting a specific output per feature vector.

Compared to their supervised kin, unsupervised algorithms are difficult to describe in full generality as their formalism depends significantly on the specific type of task they are intended for. Common tasks for such algorithms include: clustering[88] (i.e. grouping similar data points together), dimensionality reduction (discussed and exemplified in section 3.3.3), density estimation[89] (modelling the probability distribution of the data-generating process from which feature vectors are sampled), and sampling[90] (drawing synthetic feature vector examples from a learned distribution).

Semi-Supervised & Self-Supervised Learning

For some problems to which learning algorithms can be applied, difficulties arise in defining what, if anything, separates training data into input features and ground-truth targets. Algorithms operating in such contexts are difficult to characterize as either supervised or unsupervised—instead, the terms semi-supervised or self-supervised are commonly used. Although sometimes used interchangeably, they carry different connotations in the literature.

Semi-supervised learning[91] denotes methods that use both labelled and unlabelled data during the training process. A variety of techniques exist for semi-supervised learning, such as co-training[92], multi-view learning[93], and graph-based methods[94]. One popular approach is the use of consistency regularization[95], which encourages the model to produce similar predictions for different perturbations of the same input, leveraging the structure in the unlabelled data.

Self-supervised learning[96], on the other hand, involves creating auxiliary supervised learning tasks by creating artificial ground-truth targets from the raw input. By solving such tasks, the model can learn instrumental representations or features that can, for example, later be fine-tuned for the main task using (a smaller amount of) labelled data. These auxiliary tasks are designed in such a way as to be tractable using only input features, allowing the model to, as it were, learn by supervising itself. Examples of self-supervised learning tasks include predicting the next word in a sentence[97], or predicting a masked part of an image[98].

Reinforcement Learning

Although not directly relevant to the work of this dissertation, *reinforcement learning* is a historically and conceptually important area of learning algorithms[99]. In this paradigm, the model to be trained (usually referred to as the *agent*) does not learn from a curated dataset, but rather learns to map situations to actions by optimizing a reward signal received from some environment.

A reinforcement learning problem is fully described by a Markov decision process, defined as a tuple $\mathcal{W} = (\mathbb{S}, \mathbb{A}, P, R, \gamma)$, where $\mathbb{S}$ is the state space (the set of possible world configurations) and $\mathbb{A}$ the action space (the set of actions the agent can perform). The state transition probability function $P = P(s, s', a) \in [0, 1]$ defines the probability of transitioning from state $s \in \mathbb{S}$ to $s' \in \mathbb{S}$ if action $a \in \mathbb{A}$ is taken. $R = R(s, s', a) \in \mathbb{R}$ is the reward function, defining the reward (if positive) or punishment (if negative) obtained if the agent transitions from state s to state s' via action a. Lastly, γ is the discount factor, representing the the relative value of rewards obtained sooner rather than later.

From these foundational constructs, the concept of a *policy* can be defined. A policy, denoted $\pi : \mathbb{S} \rightarrow \mathbb{A}$, is a mapping from states to actions. Broadly construed, the goal of reinforcement learning is to learn an optimal (conditional on $\mathcal{W}$) policy function, where optimality entails maximizing the (expected) accumulated reward over time. Two main classes of methods exist for this purpose: value-based and policy-based.

Value-based methods, such as Q-learning[80] and SARSA[100], learn to approximate either the optimal value function $V^*(s)$ or the optimal state-value function $Q^*(s, a)$. $V^*(s)$ returns the expected cumulative future reward from a given state s, given an optimal policy. Similarly, $Q^*(s, a)$, returns the expected cumulative future reward from taking action a in state s under optimal policy. With V^* and/or Q^* known, an optimal policy can be derived, for example by computing $\pi^*(s) = \operatorname{argmax}_a Q(s, a)$.

Policy-based methods, like policy gradient[101] and REINFORCE[102], approximate the optimal policy π^* directly. This is achieved by optimizing the free parameters of some direct model of the policy function w.r.t. expected future reward. While simple to state, the algorithms used for this end are seldom completely straightforward to implement. Interested readers should consult a textbook on the subject (e.g. [99]) rather than, say, a dissertation in a barely related field.

3.3 Algorithms

This section contains an account of a selection of concrete tasks (directly or indirectly relevant to the work presented in the papers of this dissertation) with appertaining ML algorithms popularly deployed for solving them.

3.3.1 Classification

In classification, unambiguously a form of supervised learning, the goal of the learning algorithm is to map d-dimenssional input feature vectors $\boldsymbol{X} = (\boldsymbol{x}^1, \boldsymbol{x}^2, ..., \boldsymbol{x}^n)$ to corresponding categorical targets $(\boldsymbol{y}^1, \boldsymbol{y}^2, ..., \boldsymbol{y}^n)$, where $\boldsymbol{x}^i \in \mathbb{R}^d$ and $\boldsymbol{y}^i \in \{1, ..., c\}^l$. As before, the function from inputs to outputs learned by the model during training is denoted f. In the case of *binary classification* ($c = 2$), there are two possible classes wheras in the case of *multi-class classification* ($c > 2$), there are more than two classes. In the case of *multi-output classification* ($l > 1$), each feature vector simultaneously belongs in l classes. The special case of binary multi-output classification ($c = 2, l \geq 2$) is often refered to as *multi-label classification*[103]. Although it is possible to conceive of multi-output classification problems wherein each of the l outputs have a variable number of classes $(c^1, ..., c^l)$, this case lies outside the scope of this brief review.

K Nearest Neighbours (kNN)

The K Nearest Neighbours (kNN) algorithm[104] is a popular ML architecture for single-output classification, well-known for its simplicity (conceptual, not computational) and complete transparency. It has a single hyperparameter, k, which must be selected in advance of running the model.

The training process of kNN is the following: given a training dataset, the kNN algorithm simply stores it in its entirety in memory. Thus, the free parameters θ of kNN are exactly equivalent to the training dataset $\mathbb{D}$.

During inference, kNN estimates the category $\hat{y}$ of a novel input feature vector $\boldsymbol{x}'$ by finding the k closest feature vectors in the memorized training dataset. The output value is then set to the mode of the classes of these k nearest neighbours:

$$\hat{y} = f(\boldsymbol{x}') = \underset{j}{\operatorname{argmax}} \sum_{i \in N_k(\boldsymbol{x}')} \mathbb{I}(y^i = j) \qquad \text{(eq. 3.1)}$$

where $y^i \in \{1, ..., c\}$ is the class of the ith training example and $N_k(\boldsymbol{x}')$ is the set of indices of the k nearest neighbours of $\boldsymbol{x}'$ among the feature vectors of the training data.

Although the simple kNN inference rule in eq. 3.1 only functions for single-output ($l = 1$) classification (as the equality would be ill-defined with vector-valued $\boldsymbol{y}^i$), extensions of the algorithm that allow for deployment in multi-output contexts exist[105].

For the purpose of constructing the set of nearest neighbours, the distance between a canidate feature vector $\boldsymbol{x}'$ and a stored feature vector $\boldsymbol{x}_j$ can be calculated using different distance metrics. Whereas the Euclidean distance $\sqrt{\|\boldsymbol{x}' - \boldsymbol{x}^j\|_2}$ is the most common, alternatives such as the Manhattan distance and the Minkowski distance can be used[106].

Linear discriminant analysis (LDA)

Linear Discriminant Analysis (LDA)[107] is a learning algorithm for multi-class, single-output classification. Informally, it based on the concept of projecting a multi-dimensional dataset onto a lower-dimensional space where discrimination between classes is maximally possible. Concrelely. the projection is chosen in such a way as to maximize the between-class variation while minimizing the within-class variation. While the algorithm assumes that the class-conditional probability densities $\{Pr(\boldsymbol{x}|y = 1), ..., Pr(\boldsymbol{x}|y = c)\}$ are Gaussian and have equal covariance matrices, it often functions adequately even when these conditons are not met[108].

Using the same notation as previously, the main idea behind LDA is to find the transformation matrix $\boldsymbol{W} \in \mathbb{R}^{(c-1)\times d}$ that linearly projects a feature vector $\boldsymbol{x}$ onto a $(c-1)$-dimensional space, such that the class separation is maximized.

During training, the optimal projection is achieved by maximizing Fisher's criterion $J(\boldsymbol{W}) \in \mathbb{R}$, defined by:

$$J(\boldsymbol{W}) = \frac{\det(\boldsymbol{W}^T \cdot \boldsymbol{S}_B \cdot \boldsymbol{W})}{\det(\boldsymbol{W}^T \cdot \boldsymbol{S}_W \cdot \boldsymbol{W})} \tag{eq. 3.2}$$

where $\boldsymbol{S}_B$ is the between-class scatter matrix and $\boldsymbol{S}_W$ is the within-class scatter matrix, defined as:

$$\boldsymbol{S}_B = \sum_{i=1}^{c} n^i (\mu_i - \mu)(\mu_i - \mu)^T, \tag{eq. 3.3}$$

$$\boldsymbol{S}_W = \sum_{i=1}^{c} \sum_{\boldsymbol{x} \in \mathbb{C}^i} (\boldsymbol{x} - \mu_i)(\boldsymbol{x} - \mu_i)^T, \tag{eq. 3.4}$$

with μ^i being the mean vector for class i, μ representing the overall mean vector, and n^i denoting the number of samples in class i. $\mathbb{C}^i$ denotes the set of training feature vectors belonging to the ith class. To obtain the optimal projection matrix $\boldsymbol{W} = \operatorname{argmax}_{\boldsymbol{W}} J(\boldsymbol{W})$, one can simply solve the generalized eigenvalue problem:

$$\boldsymbol{S}_W^{-1} \cdot \boldsymbol{S}_B \cdot \boldsymbol{w}^i = \lambda^i \boldsymbol{w}^i, \tag{eq. 3.5}$$

where $\boldsymbol{w}^i$ and λ^i are eigenvectors and eigenvalues, respectively. The rows of $\boldsymbol{W}$ are formed by the $(c-1)$ eigenvectors corresponding to the largest eigenvalues.

During inference, LDA performs classification via discriminant functions. Given a new sample $\boldsymbol{x}'$, its class membership is determined by discriminant functions $\{g^1, \ldots, g^c\}$:

$$g^j(\boldsymbol{x}') = \boldsymbol{W}_{i,*} \cdot \boldsymbol{S}_W^{-1}(\boldsymbol{x} - \mu^i), \tag{eq. 3.6}$$

The sample is subsequently assigned to the class with highest discriminant value:

$$\hat{y} = f(\boldsymbol{x}) = \operatorname*{argmax}_i g^j(\boldsymbol{x}') \tag{eq. 3.7}$$

Although producing only a single output per feature vector in the above framework, LDA can be extended to multi-output classification in a multitude of ways[109].

Support Vector Machines (SVM)

The Support Vector Machine (SVM) algorithm[110] is a learning algorithm for single-output, binary classification. To make notation less dense, binary target variables are here coded as $y^i \in \{-1, 1\}$ rather than $y^i \in \{0, 1\}$ as previously.

SVM seeks to find the optimal hyperplane that separates feature vectors of two different classes with the maximum margin. Let $\boldsymbol{w}^T\boldsymbol{x} + b = 0$ be the equation for the optimal hyperplane, where $\boldsymbol{w} \in \mathbb{R}^d$ is the learnable weight vector and $b \in \mathbb{R}$ is the learnable bias term. The width of the widest possible margin between the two classes can then be computed as $\frac{2}{|\boldsymbol{w}|}$, and the optimization problem to be solved as:

$$\min_{\boldsymbol{w},b} \quad \frac{1}{2}|\boldsymbol{w}|\hat{}2 \quad \text{subject to} \quad y^i(\boldsymbol{w}^T\boldsymbol{x}^i + b) \geq 1 \forall i \qquad \text{(eq. 3.8)}$$

This is a convex quadratic programming (QP) problem with linear constraints, which can be solved by formulating the Lagrangian dual problem. While not reproduced here, it can be efficiently solved to obtain the optimal weight vector $\boldsymbol{w}$ and bias b using quadratic programming techniques.

For inference, the SVM classifier can make predictions for a new data point $\boldsymbol{x}' \in \mathbb{R}^d$ by evaluating the decision function $f\colon \mathbb{R}^d \to -1, 1$.

$$\hat{y} = f(\boldsymbol{x}') = sgn(\boldsymbol{w} \cdot \boldsymbol{x}' + b) \qquad \text{(eq. 3.9)}$$

The aforementioned version of SVM is typically called linear SVM, as it only functions when a linear decision boundrary is sufficient. A simple modification exists that largely remedies this shortcoming. The so-called *kernel trick* is a technique used to enable SVM to find non-linear decision boundaries by implicitly transforming the input data into a higher-dimensional space. It achieves this by replacing the dot product in the feature space in eq. 3.9 with a kernel function, which computes the inner product in the transformed space without explicitly performing the transformation. By using the kernel trick, SVM can learn many complex, non-linear decision boundaries without increasing the computational complexity significantly[111, 112].

In most non-toy problems, it is common to have training data with overlapping classes, where perfect separation of the classes is impossible even after applying a nonlinear kernel transformation. In such cases, the problem in eq. 3.8 becomes unsolvable, as no separating hyperplane exists. Fortunately, SVM can still be used effectively by allowing some misclassifications, which is achieved by introducing slack variables and using a soft-margin approach (see [110] for full description).

For multi-class problems with $c > 2$ classes, SVM (and any other binary classifier) can be extendedrelatively easily by using either a one-vs-one (OVO) or one-vs-rest (OVR) strategy[113]. In OVO, $\frac{c(c-1)}{2}$ binary SVM classifiers are trained, each separating one class from another. For a new data point $\boldsymbol{x}'$, each classifier votes for a class, and the class with the most votes is assigned as the final prediction. In OVR, c binary SVM classifiers are trained, each separating one class from the rest. The classifier with the highest confidence, measured by the distance to the decision boundary, determines the class label for $\boldsymbol{x}'$.

3.3.2 Regression

Regression is a class of supervised learning problems wherein output targets are continuous rather than categorical. Formally, the goal of the learning algorithm is to map d-dimensional input feature vectors $\boldsymbol{X} = (\boldsymbol{x}^1, \boldsymbol{x}^2, ..., \boldsymbol{x}^n)$ to corresponding targets $\boldsymbol{Y} = (\boldsymbol{y}^1, \boldsymbol{y}^2, ..., \boldsymbol{y}^n)$, where $\boldsymbol{x}^i \in \mathbb{R}^d$ and $\boldsymbol{y}^i \in \mathbb{R}^l$. Depending on context, the feature vectors that make up $\boldsymbol{X}$ are referred to as *predictors*, *regressors*, or *independent variables*, whereas the target values that make up $\boldsymbol{Y}$ are referred to as *regressands* or *dependent variables*.

Like with classification, architectures for regression can be divided into those with a single scalar-valued target ($l = 1$, often called simple regression) and those with a vector-valued target ($l > 1$, often called multiple regression). Regression methods can be further subdivided into those that learn to perform a linear mapping (section 3.3.2) versus those that do not (section 3.3.2).

Linear Regression

Linear regression is a family of regression models for which model output is linear w.r.t model input, i.e. $f(\boldsymbol{a} + \boldsymbol{b}) = f(\boldsymbol{a}) + f(\boldsymbol{b})$ and $f(\lambda \cdot \boldsymbol{x}) = \lambda \cdot f(\boldsymbol{x})$ [114]. Although most likely already familiar to the reader, its underpinnings are reproduced here to provide notation to be extended for the nonlinear case in the next section.

By definition, the mapping from a input feature vector $\boldsymbol{x} \in \mathbb{R}^d$ to output $\hat{\boldsymbol{y}} \in \mathbb{R}^l$ performed by a linear regression model can be stated as:

$$\hat{\boldsymbol{y}} = f(\boldsymbol{x}) = \boldsymbol{W} \cdot \boldsymbol{x} \qquad \text{(eq. 3.10)}$$

Where $\boldsymbol{W} \in \mathbb{R}^{l \times d}$ is a learnable affine transformation matrix. The simplest training objective used for linear regression is to minimize the residual sum of squares loss function:

$$\mathcal{L}(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{W}) = RSS = \sum_{i=1}^{n} \sum_{j=1}^{l} (\boldsymbol{Y}_{i,j} - f(\boldsymbol{X}_{i,*})_j)\hat{}2 \qquad \text{(eq. 3.11)}$$

The unique optimal solution $\boldsymbol{W}^*$ can then be computed via:

$$\boldsymbol{W}^* = \underset{\boldsymbol{W}}{\operatorname{argmin}}\, RSS = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y} \qquad \text{(eq. 3.12)}$$

For other choices of loss function (i.e. sum of absolute differences), there might not exist a closed-form solution. Furthermore, for large training datasets, it might not be tractable to compute the inverses in eq. 3.12 due to memory constraints. In such cases, an approximately optimal solution can be found through iterative methods. For any choice of loss function of the form $\mathcal{L}(\boldsymbol{y} - \hat{\boldsymbol{y}})$, where $\mathcal{L}$ is convex, iterative local methods are guaranteed to converge, as the underlying optimization problem will be convex as well[115].

Nonlinear Regression

Unlike linear regression, which assumes a linear relationship between variables, nonlinear regression can fit a wider variety of relationships. Any nonlinear mapping can equivalently be represented by applying a (parametrized) nonlinear transformation $\phi : \mathbb{R}^d \times \mathbb{P} \rightarrow \mathbb{R}^k$, to the input variables, where $\mathbb{P}$ is the parameter space of the transformation, followed by linearly projecting the transformed inputs onto the outputs using a weight matrix $\boldsymbol{W} \in \mathbb{R}^{l \times k}$[85], i.e.:

$$\hat{\boldsymbol{y}} = f(\boldsymbol{x}; \boldsymbol{\theta}, \boldsymbol{W}) = \boldsymbol{W} \cdot \phi(\boldsymbol{x}; \boldsymbol{\theta}) \qquad \text{(eq. 3.13)}$$

Like in the linear case, the regression problem can be formulated as a search problem, where the goal is to find the nonlinear function $f(\boldsymbol{x}; \boldsymbol{\theta}, \boldsymbol{W})$, from some space of candidates, that best describes the relationship between the dependent and independent variables. Here, $\theta \in \mathbb{R}^p$ is a vector of tunable parameters that must be estimated from the data. The space of possible nonlinear transformations $\mathbb{F}$, where $\phi \in \mathbb{F}$, can take various forms (e.g. polynomials, exponentials, logarithms, or compositions thereof)[116]. The entirety of chapter 4 is reserved for discussing a particular family of parametrized nonlinear mappings, namely artificial neural networks.

To quantify the quality of any given choice of free parameters $(\boldsymbol{\theta}, \boldsymbol{W})$ for the model, a loss function $L(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta}, \boldsymbol{W})$ is typically used. A possible objective during training

is to minimize this loss function with respect to the tunable parameters. Unlike linear regression, however, closed-form solutions for the optimal parameters are generally not available, necessitating the use of iterative optimization algorithms.

A prevalent optimization method for nonlinear regression training is the Levenberg-Marquardt algorithm[117]. The algorithm iteratively updates the parameter estimates as follows:

$$\theta_{k+1} = \theta_k - (\boldsymbol{J}_k^T \boldsymbol{J}_k + \lambda_k I)^{-1} J_k^T r_k \qquad \text{(eq. 3.14)}$$

where k denotes the iteration number, J_k is the Jacobian matrix of the residuals $r_k = (y_1 - f(x_1, \theta_k), \ldots, y_n - f(x_n, \theta_k))^T$, and λ_k is a damping factor—a hyperparameter of the training process itself. Many other optimization techniques, such as quasi-Newton methods, can also be employed for solving nonlinear regression problems[118].

3.3.3 Dimensionality Reduction

Dimensionality reduction denotes the unsupervised learning task of transforming some unlabelled dataset of feature vectors $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n)$ to a lower-dimensional representation $\boldsymbol{Z} = (\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_n)$, where $\boldsymbol{x}_i \in \mathbb{R}^d$, $f(\boldsymbol{x}_i) = \boldsymbol{z}_i \in \mathbb{R}^r$, and $r < d$. Some common reasons for undertaking this task are to reduce (i) the computational complexity of subsequent processing, (ii) storage requirements, and (iii) noise in the data, all while maintaining the key information necessary for analysis, visualization, and/or modelling[119]. Some algorithms for doing this are presented in this section, selected here their popularity in EMG processing.

Principal Component Analysis (PCA)

The primary objective of Principal Component Analysis (PCA)[120] is to transform a high-dimensional dataset into a lower-dimensional representation, while preserving as much of the original data's variation as possible. This is achieved by identifying orthogonal linear combinations of the original variables, known as principal components, which capture the maximum variance in the data.

Given the data matrix $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ with n examples/observations and d features per example, PCA aims to find a set of d pair-wise orthogonal linear combinations, $\boldsymbol{Y} = \boldsymbol{P} \cdot \boldsymbol{X}$, where $\boldsymbol{P} in \mathbb{R}^{p \times p}$ contains the principal components as its rows. The first principal component $\boldsymbol{p}_1 = \boldsymbol{P}_{1,*}$ is defined as the linear combination that maximizes the variance of the projected data:

$$\boldsymbol{p}_1 = \underset{\boldsymbol{p}}{\operatorname{argmax}} \operatorname{Var}(\boldsymbol{X} \cdot \boldsymbol{p}) \text{ such that } ||\boldsymbol{p}||\hat{}2 = 1 \qquad \text{(eq. 3.15)}$$

The remaining principal components are defined by iteratively applying the same criterion while imposing orthogonality constraints. For the k^{th} principal component, $\boldsymbol{p}_k$:

$$\boldsymbol{p}_k = \underset{\boldsymbol{p}}{\operatorname{argmax}} \operatorname{Var}(\boldsymbol{X}\boldsymbol{p}), |, ||\boldsymbol{p}||\hat{}2 = 1, \boldsymbol{p}^T \boldsymbol{p}_i = 0 \text{ for } i = 1, \ldots, k-1 \qquad \text{(eq. 3.16)}$$

To derive the principal components, the data is first centered by subtracting the mean of each variable. Then, the sample covariance matrix $\boldsymbol{S}$ can be computed via:

$$\boldsymbol{S} = \frac{1}{n-1} \boldsymbol{X} \cdot \boldsymbol{X}^T \qquad \text{(eq. 3.17)}$$

Eigenvalue decomposition of $\boldsymbol{S}$ yields the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_d$ and the corresponding eigenvectors $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_d$. The eigenvectors corresponding to the largest eigenvalues are the principal components[84], and the proportion of the total variance explained by the kth principal component is given by $\frac{\lambda_k}{\sum i=1^d \lambda_i}$.

In practice, PCA can be computed efficiently using Singular Value Decomposition (SVD) on the centered data matrix $\boldsymbol{X}$. The SVD of $\boldsymbol{X}$ is given by $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$, where $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthogonal matrices containing the left and right singular vectors, and $\boldsymbol{\Sigma}$ is a diagonal matrix containing the singular values. The principal components are given by the columns of $\boldsymbol{V}$, and the transformed, dimensionality-reduced data is obtained as $\boldsymbol{Z} = \boldsymbol{V} \cdot \boldsymbol{X}$[121].

Nonegative Matrix Factorization (NMF)

The Non-negative Matrix Factorization (NMF) algorithm[122] aims to decompose a non-negative data matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ into two lower-dimensional non-negative matrices $\mathbf{W} \in \mathbb{R}^{n \times k}$ and $\mathbf{H} \in \mathbb{R}^{k \times d}$, such that their product approximates the original data matrix, i.e., $\mathbf{X} \approx \mathbf{WH}$. Here, k is the reduced dimensionality, with $k \ll \min(m, n)$. The matrices $\mathbf{W}$ and $\mathbf{H}$ are considered as the basis and coefficient matrices, respectively. The NMF problem can be formulated as an optimization problem:

$$\min_{\mathbf{W},\mathbf{H}} \frac{1}{2} \|\mathbf{V} - \mathbf{WH}\|_F\hat{}2 \quad \text{subject to} \quad \mathbf{W} \geq 0, \mathbf{H} \geq 0 \qquad \text{(eq. 3.18)}$$

where $\|\cdot\|_F$ denotes the Frobenius norm and the inequality constraints are element-wise. NMF is a non-convex optimization problem, and various algorithms have been developed to find locally optimal solutions. One widely-used approach[123] is the multiplicative update rule, which iteratively updates $\mathbf{W}$ and $\mathbf{H}$ as follows:

$$\mathbf{W} \leftarrow \mathbf{W} \odot \frac{\mathbf{V}\mathbf{H}^T}{\mathbf{W}(\mathbf{H}\mathbf{H}^T)} \tag{eq. 3.19}$$

$$\mathbf{H} \leftarrow \mathbf{H} \odot \frac{\mathbf{W}^T\mathbf{V}}{(\mathbf{W}^T\mathbf{W})\mathbf{H}} \tag{eq. 3.20}$$

The update rules guarantee non-increasing objective function value, and convergence is achieved when a stationary point is reached.

NMF is sometimes advantageous over PCA due to its non-negativity constraints. These constraints lead to part-based representations that are easily interpretable, as the basis vectors $\mathbf{W}$ can be directly related to the original data features.

3.4 Generalization

As was exemplified throughout section 3.3, ML model training typically involves tuning a set of free parameters $\boldsymbol{\theta}$ of a function $f(\boldsymbol{x}; \boldsymbol{\theta})$ in such a way as to minimize a loss function $\mathcal{L}$ that measures the discrepancy between desired and actual outputs. Considered in isolation, this task is nothing more than a case of mathematical optimization. However, an additional desideratum exists in the field of ML: trained models should be able to *generalize*. Here, generalization denotes the ability of a model to perform well not just on examples processed during training, but on previously unseen data (sampled from the same process). However, reducing the error on the the training data does not always achieve this outcome, as models can *overfit* the training data.

The overfitting phenomenon is typically framed in terms of *excess capacity*, wher capacity denotes a model's expressive power. Informally, it represents the range of functions a model can learn and, consequently, its ability to fit various data patterns. A model with high capacity has the potential to instantiate many possible mappings between inputs and outputs, while a low-capacity model is limited to a smaller set of mappings. Formally, a model's capacity can be quantified by the Vapnik-Chervonenkis dimension or Rademacher complexity, although these measures are not always informative or computationally tractable.

Capacity is usually (but not necessarily) related to the number of free parameters in a model—as the number of free parameter increases, the *hypothesis space* $\mathbb{H}$ (the space of possible learnable functions $f(\boldsymbol{x}, \boldsymbol{\theta})$ where $f \in \mathbb{H}$) of the model grows in tandem.

Overfitting occurs when a model with high capacity learns to fit the training data too closely, capturing not only the underlying patterns but also the noise or irrelevant features of the data. As a result, the model may perform poorly on unseen data, despite having a low error on the training set. In contrast, underfitting arises when a model with low capacity is unable to capture the underlying patterns in the data, leading to high training and testing errors.

The No Free Lunch Theorem

A useful tool for understanding the inherent difficulties in getting models to generalize is a set of theorems by Wolpert *et al.* called the No Free Lunch (NFL) theorems of ML[124, 125]. One NFL theorem states that given the set of all learning algorithms $\mathbb{L}$ and the set of all data-generating distributions $\mathbb{D}$, for any two learning algorithms $A_1 \in \mathbb{L}$ and $A_2 \in \mathbb{L}$, there exists a distribution $D \in \mathbb{D}$ such that the expected performance of A_1 on D is better than than of A_2 and vice versa. This holds true even when one of the algorithms is random guessing or, say, predicting the opposite of what was learned during training. Corollary, no algorithm can be said to perform better than any other when averaged over all possible problems or data-generating distributions. This entails that, unless assumptions are made about the prior probability distribution over possible problems in $\mathbb{D}$ a model will encounter, the performance of a learning algorithm on some dataset contains no information on how it will perform on other possible inputs (illustrated in Fig. 3.1). In fact, for *any* two (different) algorithms $A \in \mathbb{L}$ and $B \in \mathbb{L}$, there exists 'as many' problems for which A outperforms B on the test set as vice versa.

While the NFL thus states, in very rough terms, that generalization is impossible in theory, it is often possible in practise. Importantly, the theorem's scope is limited to the consideration of all possible problems $\mathcal{D}$, many of which may be highly contrived or irrelevant in practice. In reality, ML optimization problems often exhibit structure or regularities that can be exploited by algorithms to perform well on novel data. For instance, in Fig. 3.1, it is more often than not safe to assume that the apparent linear relationship will continue even for previously unseen data. The properties of optimization algorithms that allow them to exploit such regularities is known as inductive biases[126]—a topic of far too extensive scope to be given justice here. Importantly, the fact that generalization happens is an empirical fact about our world and does not reflect any underlying mathematical inevitability.

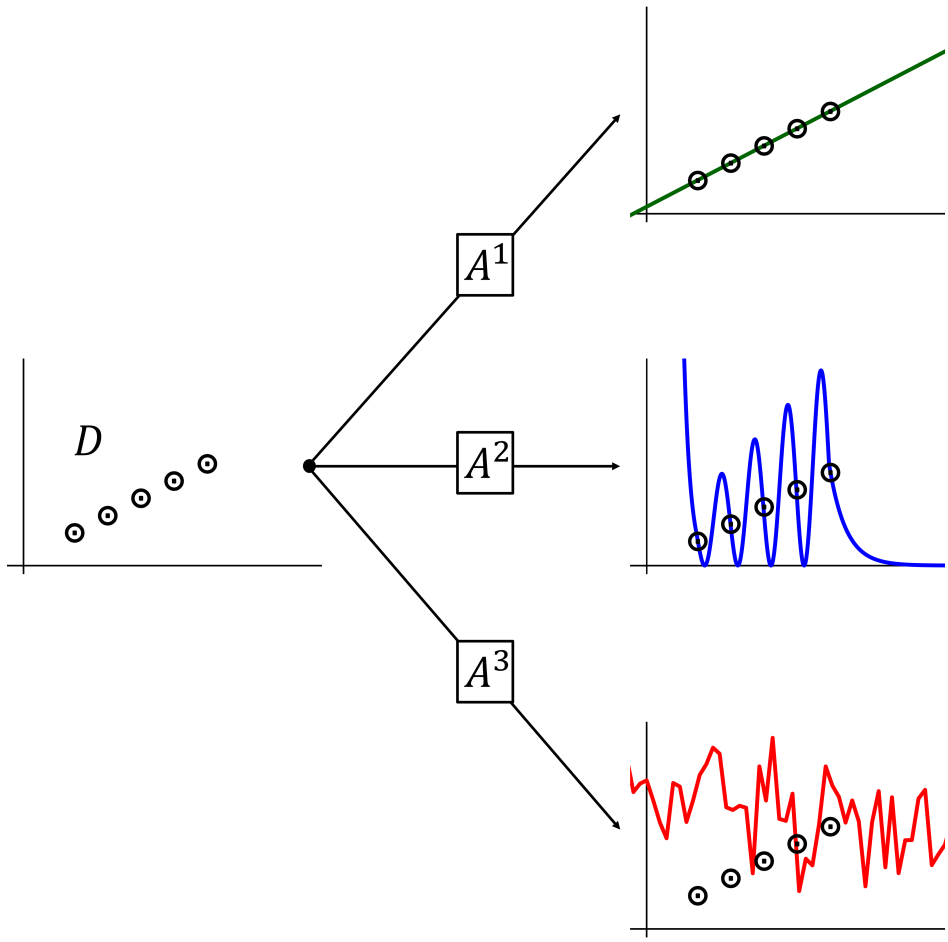


Figure 3.1: Illustratation of the NFL theorem with a training dataset sampled from a distribution D and 3 learning algorithms A^1, A^2, and A^3. A^1 and A^2 produce models that fit the training data exactly using linear regression and some complicated parametrized function, respectively. A^3 completely disregards the training data and produces a model that guesses randomly. Across these (and all other possible) learning algorithms, none has lower expected out-of-sample accuracy than the others, even if the expectation is only taken across all data-generating distributions compatible with the observed training set. To be able to conclude, as most of us want, that A^1 would most likely outperform at least A^3 and perhaps even A^2 in most cases, a non-uniform prior probability distributions over data-generating distributions is necessary.

Bias-Variance Tradeoff

The *bias-variance tradeoff* [127] is an empirical observation about statistical models in general. While not a formal theory that holds true for all possible learning problems, it can offer a useful framing to understand the relationship between capacity, overfitting, and underfitting. The mathematical terminology in the case of a single-input, single-output regression problem will be considered here—it can trivially be extended to other supervised inference tasks.

Let $f^* : \mathbb{R} \to \mathbb{R}$ be an unknown (deterministic[6]) function that maps input x to the target output $y = f^*(x)$. We wish to train a model $f(x; \boldsymbol{\theta})$ by tuning the free parameters $\boldsymbol{\theta}$ in such a way as to make the input-output mapping of f similar to that of f^*. For this purpose, we use a dataset $\mathbb{D}$ containing known input-output pairs $(x, f^*(x))$. The expected squared error of the learned function is then:

$$\text{Expected Test Error} = \mathbb{E}\left[(\left[f(x; \boldsymbol{\theta})\right] - f^*(x))\hat{}2\right] \tag{eq. 3.21}$$

where the expectation operator is applied over all possible training datasets $\mathbb{D}$. Given that the specific dataset used for training is independent of the target function, the expression above can be rewritten as:

$$\text{Expected Test Error} = \text{Bias}\hat{}2 + \text{Variance} \tag{eq. 3.22}$$

where

$$\text{Bias} = \mathbb{E}\left[f(x; \boldsymbol{\theta})\right] - f^*(x) \tag{eq. 3.23}$$

and

$$\text{Variance} = \mathbb{E}\left[(f(x; \boldsymbol{\theta}) - \mathbb{E}\left[f(x; \boldsymbol{\theta})\right])\right)\hat{}2\right] \tag{eq. 3.24}$$

In words, the expected error of a model on previously unseen data can be decomposed into two components: bias and variance. The bias term quantifies the model's inherent assumptions, which may not align with the true underlying patterns of the data. The variance term captures the model's sensitivity to small fluctuations in the training data.

The bias-variance tradeoff is the empirical observation that as one of these component decreases, the other often increase. High-capacity model tends to have low bias but high variance, as it is flexible enough to learn complex patterns but may overfit to the noise in the training data. Conversely, low-capacity models are typically characterized by high

bias and low variance, as it makes strong simplifying assumptions about the data that may not hold true, leading to underfitting.

The bias-variance tradeoff suggests that there typically exists an optimal model capacity that minimizes the expected test error by balancing the bias and variance components. Reducing one source of error often leads to an increase in the other, as the two are inherently interconnected. When a model is simplified to reduce variance, it becomes less sensitive to noise in the training data, but its assumptions about the underlying patterns become stronger, increasing bias. Conversely, when a model is made more complex to reduce bias, it becomes more flexible and capable of learning the true underlying patterns, but its sensitivity to noise and random fluctuations in the training data increases, leading to higher variance.

3.5 Evaluation

3.5.1 Methods for Gauging Generalization

Tied to the discussion in section 3.4, the performance of models must necessarily be assessed through their ability to generalize to unseen data in order to avoid measuring overfitted performance. To this end, several evaluation frameworks have been proposed.

Data partitioning[84] is a simple and widely used method for evaluating machine learning models. In this approach, the available data is divided into two or three disjoint sets, typically the training set, validation set, and test set. The training set is used to train the model, while the validation set is utilized for hyperparameter tuning and model selection. Finally, the test set is employed to assess the model's performance on unseen data. The sizes of these sets can vary, but the majority of the data is usually reserved for training, whereas the remaining data is allocated for validation and testing.

Cross-validation[128], sometimes called *rotation estimation*, is an evaluative framework wherein the model under consideration is retrained and retested multiple times with novel partitioning of the data. The most common form of cross-validation is k-fold cross-validation. In this technique, the data is partitioned into k equally-sized folds. The model is then trained and tested k times, with each fold serving as the test set exactly once, while the remaining $k-1$ folds are combined to form the training set. The overall model performance is then computed as the average of the performance metrics across all k iterations.

A variation of cross-validation is the leave-one-out cross-validation, where k is set equal to the number of samples in the dataset, i.e., each sample serves as a test set once. This method provides a nearly unbiased estimate of the model's performance, but can

be computationally expensive for large datasets[129].

Bootstrapping[130] is another resampling-based evaluation framework that involves drawing multiple random samples with replacement from the original dataset to create different training and test sets. A model is trained on each training set and evaluated on its corresponding test set. The average performance across all iterations provides an estimate of the model's generalization ability. Bootstrapping can be particularly useful when the available data is scarce or imbalanced.

3.5.2 Metrics

To conclude the chapter, this section presents a selection of prominent quantitative metrics used to gauge the performance of learning algorithm on unseen data.

Classification

Confusion Matrix: A confusion matrix[131] is a table used to describe the performance of a classification model. It displays the number of correct and incorrect predictions made by the classifier, organized by classes. In a multi-class problem with $c \geq 2$ classes, the confusion matrix $\boldsymbol{M} \in \mathbb{N}^{c \times c}$ has the following structure: each row represents the instances of the true class, and each column represents the instances of the predicted class (although the transpose is sometimes used in the literature). The element $M_{i,j}$ of the confusion matrix is the number of instances with true class i that were classified as class j. For multi-output problems, a confusion matrix can be created for each output separately, and the results can be combined or analyzed individually.

Accuracy: Accuracy is the ratio of the total number of correct predictions to the total number of predictions made by the classifier. Mathematically, it is defined as:

$$\text{Accuracy} = \frac{\sum_{i=1}^{c} M_{i,i}}{\sum_{i=1}^{c} \sum_{j=1}^{c} M_{i,j}} \qquad \text{(eq. 3.25)}$$

In multi-class problems, accuracy can be used as a single metric to assess the overall performance of the classifier. However, for multi-output problems, accuracy should be calculated for each output separately and then averaged or analyzed individually, as the performance of the classifier may vary for different outputs.

A limitation of accuracy is that it might mislead for imbalanced datasets, where one class is significantly more prevalent than others.

Precision: Precision[132], also known as positive predictive value, is the ratio of true positive predictions to the total number of positive predictions made by the classifier.

For the ith class, it is defined as:

$$\text{Precision}_i = \frac{M_{i,i}}{\sum_{k=1}^{c} M_{k,i}} \quad \text{(eq. 3.26)}$$

In multi-class problems, precision can be calculated for each class separately, and then averaged to obtain a single metric (macro-averaging), or weighted by class size (micro-averaging). For multi-output problems, precision should be calculated for each output and class separately, and then averaged or analyzed individually.

Specificity: Specificity, also known as the true negative rate, is the proportion of true negatives out of the total actual negatives. For the ith class, it is given by:

$$\text{Specificity}_i = \frac{\sum_{j \neq i} \sum_{k \neq i} M_{j,k}}{\sum_{j=1}^{c} \sum_{k \neq i} \left(M_{j,k}\right) - M_{i,i}} \quad \text{(eq. 3.27)}$$

In multi-class problems, specificity is calculated for each class separately and can be averaged using macro- or micro-averaging. In multi-output problems, specificity is calculated for each output variable separately. Specificity is useful when the cost of false positives is high.

Sensitivity: Sensitivity[132], also known as recall or true positive rate, is the ratio of true positive predictions to the total number of instances in the true class. For the ith class, it is defined as:

$$\text{Recall}_i = \frac{M_{i,i}}{\sum_{k=1}^{c} M_{i,k}} \quad \text{(eq. 3.28)}$$

As with precision, recall can be calculated for each class separately in multi-class problems and then averaged (macro-averaging) or weighted by class size (micro-averaging). For multi-output problems, recall should be calculated for each output and class separately, and then averaged or analyzed individually.

F1 Score: The F1 score[132] is the harmonic mean of precision and recall, providing a single metric that balances both values. It is particularly useful when dealing with imbalanced datasets or when both false positives and false negatives are equally important. The F1 score is defined as:

$$\text{F1 Score}_i = 2 \cdot \frac{\text{Precision}_i \cdot \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i} \quad \text{(eq. 3.29)}$$

Regression

All regression performance metrics presented in this section can be directly applied to multi-output regression problems, where each target variable can be treated independently. To obtain an overall measure of model performance, these metrics can be calculated for each target variable, and then averaged or aggregated in some manner, such as taking the mean or the weighted average of the per-output metrics.

Mean Absolute Error (MAE): Mean Absolute Error[133] is a measure of the average magnitude of errors in a set of predictions, without considering their direction. It is defined as the average of the absolute differences between the predicted values and the true values. Mathematically, given n data points, true values y_i, and predicted values $\hat{y}_i$, the MAE is defined as:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \qquad \text{(eq. 3.30)}$$

Root Mean Squared Error (RMSE): Root Mean Squared Error[133] is a measure of the average squared differences between the predicted values and the true values. It is more sensitive to large errors than the MAE, as the squared differences amplify the impact of large errors. Mathematically, given n data points, true values y_i, and predicted values $\hat{y}_i$, the RMSE is defined as:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)\hat{}2} \qquad \text{(eq. 3.31)}$$

Variance Accounted For (VAF): Variance Accounted For[134] is a measure of the proportion of the total variance in the true values that is explained by the model. It is also known as the coefficient of determination (R^2) in the case of linear regression. VAF is defined as:

$$\text{VAF} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum i = 1^n(y_i - \bar{y})^2} \qquad \text{(eq. 3.32)}$$

where $\bar{y}$ is the mean of the true values. A VAF of 1 indicates that the model perfectly predicts the true values, while a VAF of 0 indicates that the model is no better than predicting the mean of the true values.

Nothing is particularly hard if you divide it into small jobs.

Henry Ford

Chapter 4

Deep Learning

Classical learning algorithms of the type introduced in the previous chapter—in addition to being of illustrative value—work exceedingly well on a number of important problems. However, they typically struggle in contexts where the input consists of high-dimensional, unstructured data[135]. Notably, many tasks necessary for 'human-level' perception and cognition—recognition of objects in images, transcriptions of sound recordings, extracting meaning from natural language, *etc.*—are contained in this class. More relevant for this dissertation, the raw voltage values that constitute electromyographic signals (described in section 2.5) are often high-dimensional and arguably unstructured. The study and development of large Artificial Neural Networks (ANNs) as a model class arose in some part out of a desire for learning algorithms that can handle problems with such properties[85].

ANNs consist of interconnected nodes (or, synonymously, neurons) that learn to represent data through progressive layers of abstraction. The titular buzzword *Deep Learning* (DL) has come to denote learning algorithms where the underlying architecture $f : \mathbb{X} \to \mathbb{Y}$ from input $\boldsymbol{x} \in \mathbb{X}$ to output $\hat{\boldsymbol{y}} \in \mathbb{Y}$ can be represented as a composition of L subtransformations, where L is 'large'[136]. This layered approach enables the learning of hierarchical representations from raw data—contrasted by classical methods, where feature vectors provided to the model must to a significant degree already capture the factors of variation that are relevant to the learning problem[85]. This structure explains the other names for the same (loosely speaking) framework: *hierarchical learning*, *feature learning*, or *representation learning*[31].

In this chapter, section 4.1 introduces ANNs as a type of model architectures and briefly discusses its roots. Section 4.2 reviews some of the ways such models can be trained to fit data. Section 4.3 reviews some model architectures of special importance to this dissertation. Section 4.4 lists techniques for combating overfitting during training. Section 4.5 some recently introduced techniques that aid in the training of truly deep ANNs. Lastly, section 4.6 seeks to illuminate the gap between theory and empirical results of ANNs by reviewing some conjectures to this end.

4.1 Artificial Neural Networks

As the name suggest, an ANN can be any model that mimics any system of biological neurons[137]. In this dissertation (and large swathes of the contemporary literature[138]), the term ANN denotes a particular subtype of such models: a parametrized function f that can be represented exactly as a combination (e.g. composition, summation, scaling, *etc.*) of a specific, simplified type of *artificial neuron*. The underlying biological inspiration for such systems, while interesting from an historical perspective, is perhaps best understood as a superficial curiosity here. After all, a skilled constructor of aeroplanes need not be an expert in ornithology.

Artificial Neurons

The artificial neuron[139] is a simple mathematical object in the form of a scalar function $\nu : \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}$, parametrized by a weight vector $\boldsymbol{w} \in \mathbb{R}^d$ and a bias $b \in \mathbb{R}$, that takes a vector-valued input $\boldsymbol{x} \in \mathbb{R}^d$ and return a single *activation* $a \in \mathbb{R}$:

$$a = \nu(\boldsymbol{x}; \boldsymbol{w}, b) = \phi(z) \quad \text{where} \quad z = b + \sum_{i=1}^{d} w_i \cdot x_i \qquad \text{(eq. 4.1)}$$

Here, $\phi : \mathbb{R} \to \mathbb{R}$ denotes a (prespecified) *activation function*.

The similarity between this abstraction and a biological neuron is most clear in the case when (i) the step function is used as the activation function, i.e. $\phi(z) = H(z)$, and (ii) a binary input vector is presumed, i.e. $\boldsymbol{x} \in \{0, 1\}^d$. In this case, the analogy is the following: each input x_i represent the firing (or lack thereof) of the ith synaptic input and $|w_i|$ represents the strength of the ith synapse, where $sgn(w_i) = 1$ corresponds to an excitatory synapse and $sgn(w_i) = -1$ to an inhibitory synapse. Lastly, $-b$ represents the membrane threshold potential. In this case, the output activation is $\nu(\boldsymbol{x}) = 1$ (a firing event) if and only if $z \geq 0$, i.e. $\boldsymbol{w} \cdot \boldsymbol{x} \geq -b$. While this is a simplified view of how biological neurons function, one must keep in mind that the purpose of artificial neurons is not to model biology, but rather to serve as building blocks for DL models.

Activation functions need to be *nonlinear* and *continuously differentiable* (for reasons outlined in sections 4.1 and 4.2, respectively) in order to be suitable for use in deep ANN models. The step function $H(x)$ used above, albeit nonlinear, is not differentiable. Common alternatives include the sigmoid activation function $\sigma(x)$ and its close relative tanh—these can be viewed as smooth variants of $H(x)$[140]:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \quad \text{and} \quad \tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} = 2\sigma(z) - 1 \qquad \text{(eq. 4.2)}$$

In contemporary DL systems, however, these vaguely biomimetic activation functions have largely been supplanted by the *rectified linear unit* (ReLU)[141]. In addition to being more compute-efficient, ReLU has the desirable property that, in contrast to σ and tanh, its repeated application on an input does not tend to zero.

$$\mathrm{ReLU}(z) = \max(0, z) \tag{eq. 4.3}$$

The success of RelU has spawned a family of similar, wedge-shaped activations functions with advantages[142] over the original—examples include ELU[143], GELU[144], and leakyReLU[145]:

$$\mathrm{ELU}(z) = \begin{cases} z \text{ if } z > 0 \\ (e^z - 1) \text{ if } z \leq 0 \end{cases} \tag{eq. 4.4}$$

$$\mathrm{GELU}(z) = x\Phi(x) \approx 0.5z(1 + \tanh[\sqrt{\frac{2}{\pi}}(z + 0.04472z^3)]) \tag{eq. 4.5}$$

$$\mathrm{leakyReLU}(z) = \max(\alpha z, z) \tag{eq. 4.6}$$

where $\Phi(x)$ is the Gauss error function and $\alpha \in [0, 1]$ is a leakage hyperparameter.

Organization

For reasons related to simplicity and ease of optimization (and with some roots in biology[146]), artificial neurons in ANNs of the type under consideration here are typically organized in *layers*. A layer denotes a subfunction $f^{(l)} : \mathbb{R}^a \to \mathbb{R}^b$ (where $f = f^{(l)} \circ ... \circ f^{(1)}$ is the function performed by the ANN model as a whole) whose artificial neurons can process their inputs in parallel without requiring the output of other neurons in the same layer. A layer is said to be *feed-forward* if each neuron receives input from the previous layer's neurons and passes the result to the next layer's nodes without any backward or recurrent connections—such a layer forms an directed, acyclic graph, where each neuron corresponds to a node[85].

The perhaps most easily definable feedforward layer is the *fully-connected layer* (also called a *dense* or *affine* layer)[84]. Such layers maps an input vector $\boldsymbol{x} \in \mathbb{R}^d$ to an output activation vector $\boldsymbol{a} \in \mathbb{R}^m$ via $m \in \mathbb{N}$ (a hyperparameter) artificial neurons that each process the entirety of the input vector:

$$a_i = \nu_i(\boldsymbol{x}; \boldsymbol{W}_{i,*}, b_i) = \phi\left(b_i + \sum_j^d \boldsymbol{W}_{i,j} \cdot \boldsymbol{x}_j\right) = \phi(z_i) \tag{eq. 4.7}$$

where $\boldsymbol{W}_{i,*}$ and b_i are the weight vector and bias, respectively, of the ith neuron. This input-output relationship is typically rewritten in more compact matrix notation:

$$\boldsymbol{a} = \mathrm{FC}(\boldsymbol{x}; \boldsymbol{W}, \boldsymbol{b}) = \phi(\boldsymbol{W} \cdot \boldsymbol{x} + \boldsymbol{b}) = \phi(\boldsymbol{z}) \qquad \text{(eq. 4.8)}$$

where the activation function ϕ is typically applied element-wise to the linearly projected outputs $\boldsymbol{z} = \boldsymbol{W} \cdot \boldsymbol{x} + \boldsymbol{b}$.

A single feedforward layer can constitute an ANN in and of itself—a so-called *single-layer perceptron*[147]—in which case the number of neurons m should be set as the number of desired outputs l. A graphical illustration of such a model is perovided in Fig 4.1. If the activation function is chosen as linear, i.e. $\phi(x) = x$, the expression in eq. 4.8 reduces to that presented in eq. 3.10. As such, a single-layered fully-connected ANN with linear activation function corresponds exactly to a linear regression model with bias[138]. By definition, such models can only learn to perform linear mappings.

A single-layer perceptron can similarly be used for single-output classification with m classes. In this case, the softmax activation function is typically used[85]. The reason for using softmax rather than a linear ϕ is related to how appropriate loss functions are defined for classification tasks (reviewed in section 4.2). In contrast to the activation functions discussed previously, softmax is not applied element-wise:

$$\boldsymbol{a} = \mathrm{softmax}(\boldsymbol{z}) \iff a_i = \frac{e^{\boldsymbol{z}_i}}{\sum_{j=1}^{m} e^{\boldsymbol{z}_j}} \qquad \text{(eq. 4.9)}$$

The output classification decision is then obtained from:

$$\hat{y} = \underset{i}{\mathrm{argmax}}\, a_i \,, \hat{y} \in \{0, ..., m\} \qquad \text{(eq. 4.10)}$$

Softmax ensures the activations are bounded ($0 < a_i < 1$) and sum to unity. As such, the activations that make up the output of softmax are often (contentiously[148]) thought of as a posterior categorical probability distribution such that, for example, a_i represents the probability of the input x belonging to class i.

A single fully connected layer with linear or softmax activation function can be found at the far end of many ANN architectures, where they exist to map the output produced by the preceding parts of the model into the form the learning problem requires. Similarly, fully connected layers are sometimes used at the very beginning of models in order do perform a (learnable) linear remapping of input data.

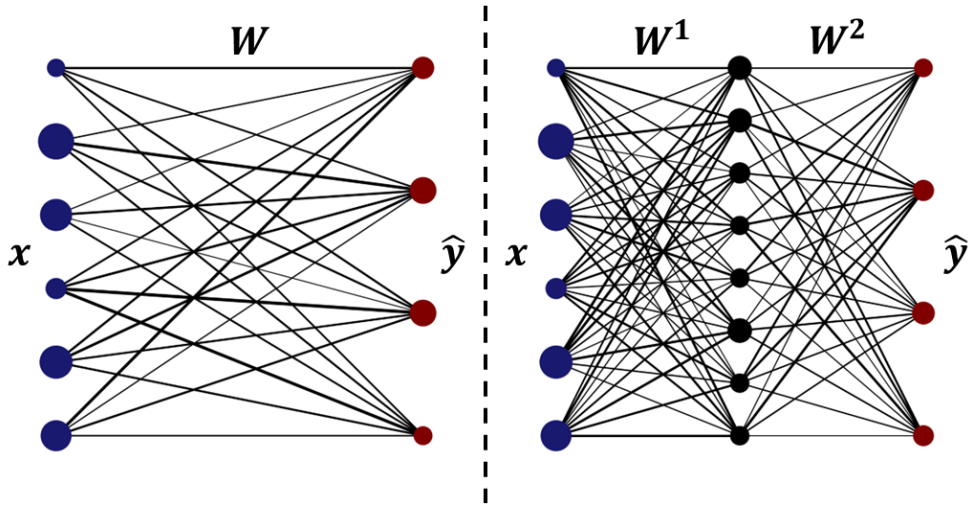


Figure 4.1: Directed acyclic graph representations of a single-layer perceptron with $d = 6$ inputs and $l = 4$ outputs (left) and a two-layer multi-layer perceptron with $d = 6$ inputs, $m = 8$ hidden neurons, and $l = 4$ outputs (right). Activations are represented by nodes with radii proportional to their magnitude. The thickness of an edge connecting node i to node j in the subsequent layer is proportional to the weight $W_{j,i}$.

Multi-Layer Perceptrons

A Multi-Layer Perceptron (MLP) is an end-to-end ANN architecture composed exclusively of (two or more) fully connected layers in sequence. A two-layer MLP can be represented as:

$$\begin{aligned} MLP(\boldsymbol{x}; \boldsymbol{W}^1, \boldsymbol{b}^1, \boldsymbol{W}^2, \boldsymbol{b}^2) &= \mathrm{FC}^2(\mathrm{FC}^1(\boldsymbol{x}; \boldsymbol{W}^1, \boldsymbol{b}^1); \boldsymbol{W}^2, \boldsymbol{b}^2) \\ &= \phi^2\left(\boldsymbol{W}^2 \cdot \phi^1(\boldsymbol{W}^1 \cdot \boldsymbol{x} + \boldsymbol{b}^1) + \boldsymbol{b}^2\right) \\ &= \hat{\boldsymbol{y}} \end{aligned} \tag{eq. 4.11}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the input vector; $\hat{\boldsymbol{y}} \in \mathbb{R}^l$ the output vector; FC^i the ith fully connected layer; $\boldsymbol{W}^1 \in \mathbb{R}^{n \times m}$ and $\boldsymbol{W}^2 \in \mathbb{R}^{m \times l}$ the weight matrices of the first and second layer; $\boldsymbol{b}^1 \in \mathbb{R}^m$ and $\boldsymbol{b}^2 \in \mathbb{R}^l$ the bias vectors of the first and second layer; and m (a hyperparameter) is the number of neurons in the first layer. ϕ^1 and ϕ^2, the activation functions of the first and second layer, respectively, where ϕ_2 is typically linear or softmax, depending on whether the output is continuous or categorical.

By convention, every layer except the final is referred to as a *hidden layer*. By iteratively adding more hidden layers, an MLP model of arbitrary depth L can be constructed:

$$f(\boldsymbol{x};\boldsymbol{\theta}) = \mathrm{FC}^{(L)}(\mathrm{FC}^{(L-1)}(\mathrm{FC}^{(L-2)}(\ldots;\boldsymbol{\theta}^{L-2});\boldsymbol{\theta}^{L-1});\boldsymbol{\theta}^{L}) \qquad \text{(eq. 4.12)}$$

where $\boldsymbol{\theta} = \{\boldsymbol{\theta}^i\}_{i=1}^{L}$ are the parameters of the model and $\boldsymbol{\theta}^i = \{\boldsymbol{W}^i, \boldsymbol{b}^i\}$ are the parameters of the ith layer. The number of neurons m^i of the ith layer (also known as the *width* of the layer), where $\boldsymbol{W}^i \in \mathbb{B}^{m_i \times m_{i-1}}$, can likewise be set as any positive integer.

Universal Approximation Theorem

Whereas single-layer perceptrons are limited to linear mappings, MLPs labour under no such restriction.The Universal Approximation Theorem[149, 150] states that a MLP with $L \geq 2$ layers can approximate *any* continuous function on any compact (i.e. closed and bounded) subset of $\mathbb{R}^n$ to an arbitrary degree of accuracy if some very lenient assumptions are satisfied—the activation functions must be continuous, non-constant, and bounded.

The theorem can be stated as follows: Let $\phi^1 : \mathbb{R} \to \mathbb{R}$ be a continuous, non-constant, and bounded activation function, let $\phi^2(z) = z$, and let $\mathbb{K} \subset \mathbb{R}^d$ be a compact set. For any continuous function $f^* : \mathbb{K} \to \mathbb{R}^l$ and any $\epsilon > 0$, there exist $m \in \mathbb{N}$, $\boldsymbol{W}^1 \in \mathbb{R}^{n \times m}$, $\boldsymbol{b}^1 \in \mathbb{R}^m$, $\boldsymbol{W}^2 \in \mathbb{R}^{m \times l}$, and $\boldsymbol{b}^2 \in \mathbb{R}^l$ such that:

$$\sup_{\boldsymbol{x} \in \mathbb{K}} \left| f^*(\boldsymbol{x}) - MLP(\boldsymbol{x}; \boldsymbol{W}^1, \boldsymbol{b}^1, \boldsymbol{W}^2, \boldsymbol{b}^2) \right| < \epsilon. \qquad \text{(eq. 4.13)}$$

In words, given any reasonable target function f^* and any desired level of maximal error $\epsilon > 0$, there exists an MLP with a single hidden layer that can approximate f^* with an error less than ϵ. The number of hidden neurons m required to achieve this level of approximation is guaranteed to be finite but naturally depends on the curvature of f^* and the desired accuracy ϵ.

The universal approximation theorem provides an important foundation to stand on for ANN models. To create more expressive ANN models, there is no need to increase the complexity of individual nodes—it is sufficient to just add more of them. However, the theorem does not state how many hidden neurons m will be needed, nor how the exact values of the free parameters (weights and biases) are be chosen to minimize the error if only a finite number of samples from f^* are available. These are the reasons for why deep ($L > 2$) MLPs , as well as architectures consisting not exclusively of fully-connected layers (some of which are discussed in section 4.3), are often needed—the universal approximation theorem by no means implies that two-layer MLPs are parameter- or sample-efficient.

4.2 Training

Although sometimes referred to as such in common parlance, MLPs (like all other ANN architectures) are not in and of themselves learning algorithms; they can more fruitfully be understood as a format for expressing arbitrary computer programs for mapping inputs to outputs[151]. In this format, the behaviour of the program is fully specified by (i) the architecture and (ii) the specific choice of free parameters. The reason for ever considering expressing programs in this (convoluted to human eyes) format is its unique amenability to *iterative improvement*.

An informal but illustrative comparison is given by Tegmark[152] (paraphrased): A program written in a text-based programming language that is changed slightly (e.g. by randomly replacing some of the tokens of the source code) is not likely to exhibit slightly different behaviour—the modified program may not even compile at all. In contrast, the output $\hat{\boldsymbol{y}}$ of an ANN model $f(\boldsymbol{x}; \boldsymbol{\theta})$ is always continuous and differentiable w.r.t. its parameters $\boldsymbol{\theta}$, meaning that a sufficiently small parameter perturbation $\Delta\boldsymbol{\theta}$ only leads to a small change $\Delta\hat{\boldsymbol{y}}$ in output. For this reason, many optimization techniques can be exploited to iteratively improve the performance of ANN models.

Training of ANN models for supervised learning tasks[7] almost[153] always involves minimizing a loss function $\mathcal{L}(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$. As in chapter 3, $\boldsymbol{X} \in \boldsymbol{R}^{n \times d}$, where $\boldsymbol{x}_i \in \mathbb{R}^d$, and $\boldsymbol{y} \in \mathbb{Y}$ are the feature vectors and targets, respectively, of a training set. As discussed in section 3.2, the target space can be either continuous, i.e. $\mathbb{Y} = \mathbb{R}^l$ (for regression problems), or categorical, i.e. $\mathbb{Y} = \{1, ..., c\}^l$ (for classification problems).

A common loss functions used in regression problems is the Mean Squared Error:

$$\mathcal{L}^{MSE}(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} (\hat{\boldsymbol{y}}_i - \boldsymbol{y}_i)\hat{}2 \qquad \text{(eq. 4.14)}$$

where $\hat{y}_i$ is the prediction of the model for the i-th data point and y_i is the corresponding ground truth.

Loss functions for classification can be separated into binary, multi-class, and multi-output classification. Many popular loss functions can be defined in terms of the Kullback-Leibler (KL) divergence[154]—a measure of dissimilarity between two probability distributions.

For binary single-output classification ($c = 2, l = 1$), the binary cross-entropy is a special case of the KL divergence between the true probability distribution $\Pr(y_i)$ and the predicted distribution $Q(\hat{y}_i)$:

$$\begin{aligned}\mathcal{L}^{BCE}(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta}) &= \frac{1}{n}\sum_{i=1}^{n} D_{KL}(P(y_i)||Q(\hat{y}i)) \\ &= -\frac{1}{n}\sum_{i=1}^{n}[y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)] \qquad \text{(eq. 4.15)}\end{aligned}$$

where $\hat{y}_i$ is the predicted probability of the positive class for the i-th data point, and y_i is the corresponding binary ground truth.

In multi-class classification, the categorical cross-entropy loss can also be seen as the KL divergence between the true probability distribution $P(\boldsymbol{y}_i)$ and the predicted distribution $Q(\hat{\boldsymbol{y}}_i)$, averaged over all data points:

$$\begin{aligned}\mathcal{L}^{CCE}(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta}) &= \frac{1}{n}\sum_{i=1}^{n} D_{KL}(P(\boldsymbol{y}\,i)||Q(\hat{\boldsymbol{y}}\,i)) \\ &= -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{c} y_{i,j}\log \hat{y}_{i,j} \qquad \text{(eq. 4.16)}\end{aligned}$$

where $\hat{y}_{i,j}$ is the predicted probability of the jth class for the ith data point, and $y_{i,j}$ is the corresponding one-hot encoded ground truth (i.e. $y_{i,j} = 1$ if the ith example belongs to the jth class and 0 otherwise).

For multi-output classification problems ($l \geq 2$) a combination (e.g. summing or weighted averaging) of output-wise cross-entropies is typically used.

Before training can commence, the initial values of the parameters $\boldsymbol{\theta}$ must be set. This process is known as *initialization*. One common initialization technique is random initialization, where the parameters are assigned random values, typically drawn from a Gaussian or uniform distribution[155]. Another somewhat more recent approach is Xavier initialization[156] (also known as Glorot initialization), which considers the size of the input and output of each layer to scale its random initial values, usually using a uniform distribution. A variation of Xavier initialization called He initialization[157] is often employed, which takes into account the specific properties of the ReLU activation function. Proper initialization helps ensure that the gradients have similar magnitudes across different layers, allowing gradient-based optimization algorithms to make more effective updates during the training process[158].

Since the introduction of trainable multi-layer architectures in the 1960s[159], various optimization methods have been applied to minimize ANN loss functions w.r.t. its free parameters, for example:

> **Evolutionary methods**[160] use natural selection-inspired mechanisms to evolve a population of solutions towards better performance. They can be used to not only to optimize the parameters of ANN models, but also to search over spaces of possible architectures in a process sometimes known as *neural architecture search*[161].
>
> **Simulated annealing**[162] is a global optimization technique that employs random sampling and a temperature-based search strategy to explore the solution space while avoiding being trapped in local minima. They have been applied to the problem of ANN loss minimization in several studies and have the advantages of being able to optimize ANNs with non-differentiable layers and/or activation functions[163].
>
> **Second-order gradient-based methods** (e.g. Newton's method, BFGS, and Levenberg-Marquardt) leverage second-order information about the curvature of the loss landscape[118]. While it is possible and arguably sometimes advantageous[164] to minimize the loss $\mathcal{L}$ of ANN models using such methods, it requires approximating the inverse of the Hessian $\boldsymbol{H}_{\mathcal{L}}^{-1} \in \mathbb{R}^{p \times p}$, where p is the number of free parameters, at every iteration during training.

Despite the great variety of possible techniques available, the by far most popular ANN training method at the time of writing is stochastic gradient descent (SGD) via *backpropagation*[165]—a computationally simple and surprisingly intelligible first-order, gradient-based method that updates parameters in a layer-wise manner.

Backpropagation

In general, *gradient descent* is a simple heuristic method for finding a (local) minima $\boldsymbol{x}^* = \operatorname{argmin}_{\boldsymbol{x}} J(\boldsymbol{x})$ of an arbitrary differentiable function $J : \mathbb{R}^m \to \mathbb{R}$[166]. The method proceeds by iteratively updating the input by moving in the direction of the negative gradient of the objective function evaluated at the previous guess $\boldsymbol{x}^{t-1}$ at step t:

$$\boldsymbol{x}^t \leftarrow \boldsymbol{x}^{t-1} - \eta \nabla J(\boldsymbol{x}) \Big|_{\boldsymbol{x}=\boldsymbol{x}^{t-1}} \qquad \text{(eq. 4.17)}$$

Where $\eta > 0$ is some (sufficiently small) update step size.

For ANN parameter search, this approach can be directly applied with minimal modifications. As the mapping from inputs and parameters to outputs performed by the model is differentiable (given that the model is a composition of differentiable layers and activation functions) and known analytically, the derivatives of $\mathcal{L}$ w.r.t. every parameter is known as well. Consequently, a parameter-wise update rule can be formulated as:

$$\theta_i^t \leftarrow \theta_i^{t-1} - \eta \frac{\partial \mathcal{L}(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta})}{\partial \theta_i}\Big|_{\theta_i = \theta_i^{t-1}} \qquad \text{(eq. 4.18)}$$

where θ_i^t is the value of the ith model parameter at step t and η is again the update step size, here called the *learning rate*—a hyperparameter that determines the size of the updates applied to the model parameters. In some formulations of the update rule, the learning rate is allowed to vary according to a prespecified schedule during the training, i.e. $\eta = \eta^t$. Typically, the entirety of the training data is not used at every step. Instead, the data is randomly split into smaller *batches* of size $b \leq n$ (a hyperparameter), each of which is sequentially used to update the model. This sampling process explains the prefix 'stochastic' in SGD.

Backpropagation denotes a particularly efficient way of computing the gradients of a layered model w.r.t. its parameters. The algorithm is based on the chain rule of calculus and consists of two steps: a forward pass and a backward pass.

During the forward pass, inputs $\boldsymbol{X}$ is passed through the network layer by layer, from the input layer to the output layer, computing the activations $\boldsymbol{a}^l$ of each layer $f^{(l)}$ as the input is transformed. At the end of the forward pass, the model produces outputs $\hat{\boldsymbol{Y}}$ corresponding to each of the inputs.

The backward pass starts at the output layer, where the loss function $\mathcal{L}$ is computed using the model outputs $\hat{\boldsymbol{Y}}$ and the ground truth targets $\boldsymbol{Y}$. The gradients of the loss function with respect to the model parameters are then computed layer by layer, moving from the output layer to the input layer. For each layer f^l, the gradient of the loss function with respect to its parameters and the gradient of the loss function with respect to its input are computed using the chain rule. Once the gradients have been computed for every parameter of every layers, the model parameters are updated as described in eq. 4.18.

The forward and backward passes, followed by the parameter update, constitute one iteration of the SGD with backpropagation. The number of steps required for the entirety of the training data to have been processed once by the network (n/b) is known as an *epoch*—as batches can be reused or resampled, optimization can proceed for any number of epochs (typically prespecified as a hyperparameter)[85].

A number of modifications of SGD have been proposed more or less specifically for DL. The arguably most straightforward of these is SGD with momentum[165], where the update equation is instead:

$$
\begin{aligned}
v_i^{t+1} &\leftarrow m \cdot v_i^t + (1-m) \cdot \frac{\partial \mathcal{L}(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta})}{\partial \theta_i}\Big|_{\theta_i = \theta_i^{t-1}} \\
\theta_i^{t+1} &\leftarrow \theta_i^t - \eta \cdot v_i^t
\end{aligned}
\qquad \text{(eq. 4.19)}
$$

where $m \in [0, 1]$ is an additional hyperparameter The addition of parameter-wise velocities v_i is indented to reduce the risk of parameters getting stuck in small, local minima or oscillate wildly during training.

Another highly impactful extension of basic SGD is known as the Adaptive Moment Estimation (Adam) optimizer[167]. Adam combines the ideas of momentum and adaptive learning rates to achieve faster convergence . The algorithm maintains separate moving averages of the gradients and the squared gradients for each model parameter, which are used to adjust the learning rate individually for each parameter during the optimization process. In more compact vector notation, the corresponding update rule is:

$$
\begin{aligned}
\boldsymbol{m}^{t+1} &\leftarrow \beta_1 \boldsymbol{m}^t + (1-\beta_1) \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta})\Big|_{\boldsymbol{\theta} = \boldsymbol{\theta}^t} \\
\boldsymbol{v}^{t+1} &\leftarrow \beta_2 \boldsymbol{v}_t + (1-\beta_2)(\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta})\Big|_{\boldsymbol{\theta} = \boldsymbol{\theta}^t})\hat{}2 \\
\hat{\boldsymbol{m}}^{t+1} &= \frac{\boldsymbol{m}^{t+1}}{1-\beta_1\hat{}t} \\
\hat{\boldsymbol{v}}^{t+1} &= \frac{\boldsymbol{v}^{t+1}}{1-\beta_2\hat{}t} \\
\boldsymbol{\theta}^{t+1} &\leftarrow \boldsymbol{\theta}^t - \eta \frac{\hat{\boldsymbol{m}}_t}{\sqrt{\hat{\boldsymbol{v}}_t} + \epsilon}
\end{aligned}
\qquad \text{(eq. 4.20)}
$$

where $\boldsymbol{m}_t$ and $\boldsymbol{v}_t$ are the moving averages of the gradients and squared gradients, respectively, at time step t, β_1 and β_2 are the exponential decay rates for the moving averages (typically set to 0.9 and 0.999, respectively), $\hat{\boldsymbol{m}}_t$ and $\hat{\boldsymbol{v}}_t$ are bias-corrected estimates of $\boldsymbol{m}_t$ and $\boldsymbol{v}_t$, η is the learning rate, and ϵ is a small constant (typically set to 10^{-8}) added for numerical stability. The power operator is denoted by $\hat{}$.

4.3 Special Architectures

4.3.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs)[168, 169] are a class of feedforward ANN models closely related to MLPs. CNNs have been found to be suitable for processing grid-like data structures such as images (which is also the original intention behind their creation), and have seen widespread success in the field of computer vision, where famous examples include AlexNet[170], VGG[171], and ResNet[136]. The key building blocks of CNNs are convolutional layers and, optionally, pooling layers and fully connected layers.

Convolutional Layers

A convolutional layer (Fig 4.2 is a feedforward layer that aims to exploit local structure in the input data by performing convolutions with a set of learnable *filter kernels*. Whereas convolutional layers can be constructed to process input data of arbitrary dimensionality, the formalism in this section will assume a two-dimensional grid for the sake of simpler notation.

Assume the input feature vector is structured as a grid (i.e. image) of width w and height h where every grid point (i.e. pixel) is described by c channels, i.e. $\boldsymbol{x} \in \mathbb{R}^{h \times w \times c}$ where $w \cdot h \cdot c = d$. The convolution operation can then be expressed as:

$$\begin{aligned} &\boldsymbol{a} = \text{Conv}(\boldsymbol{x}; \boldsymbol{K}, \boldsymbol{b}) \iff a_{\xi,\psi,\kappa} = \phi(z_{\xi,\psi,\kappa}), \\ &\text{where } z_{\xi,\psi,\kappa} = b_\kappa + \sum_{i=1}^{\mathcal{H}} \sum_{j=1}^{\mathcal{W}} \sum_{k=1}^{c} K_{i,j,k,\kappa} \cdot x_{\xi+i-1,\psi+j-1,k} \end{aligned} \qquad \text{(eq. 4.21)}$$

Here, $\boldsymbol{K} \in \mathbb{R}^{\mathcal{H} \times \mathcal{W} \times c \times \mathcal{K}}$ is a learnable kernel tensor, $\boldsymbol{b} \in \mathbb{R}^{\kappa}$ is a learnable bias vector, and $\boldsymbol{a} \in \mathbb{R}^{(h-\mathcal{H}) \times (w-\mathcal{W}) \times \mathcal{K}}$ is the output volume, where the slice $\boldsymbol{a}_{*,*,k}$ is called the kth *activation map*. Like in the fully connected layer, ϕ denotes an activation function, applied element-wise.

A convolutional layer has 3 hyperparameters: the kernel height $\mathcal{H} \in \mathbb{N}$, the kernel width $\mathcal{W} \in \mathbb{N}$, and the number of kernels $\mathcal{K} \in \mathbb{N}$. Two additional hyperparameters, called the strides s_y and s_x, are sometimes used in the literature[171] and require some modification of eq. 4.21.

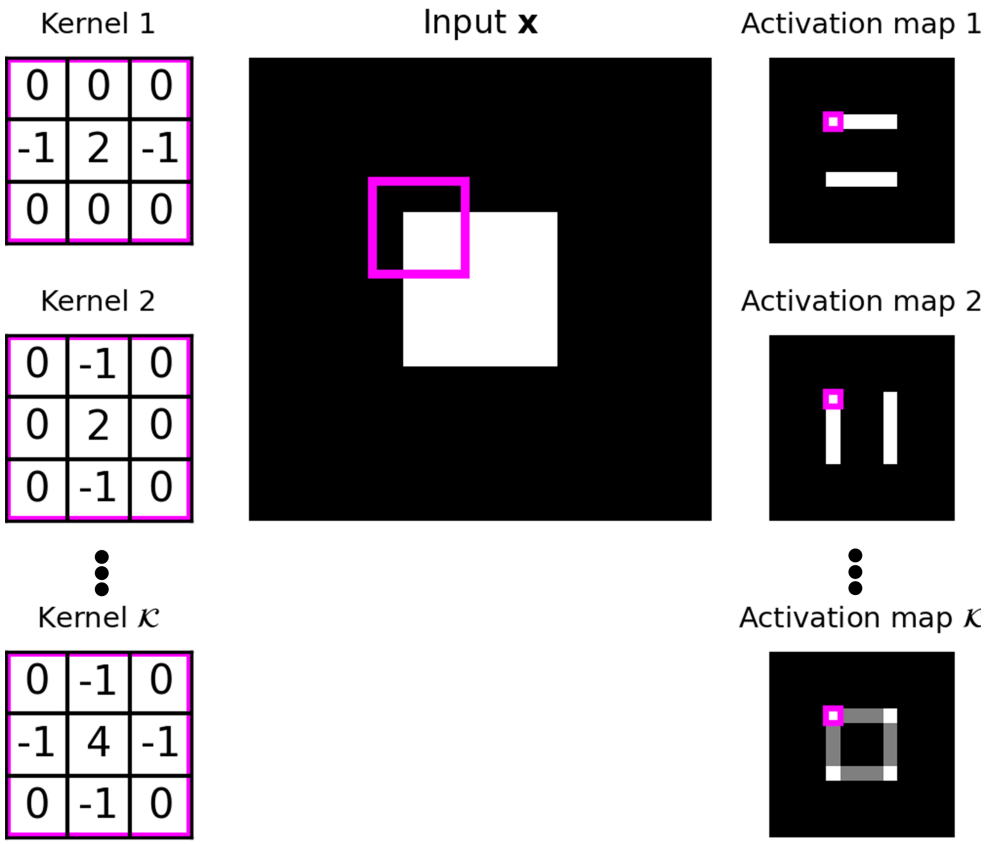


Figure 4.2: Example of a convolutional layer operating on an input grid with $c = 1$ channels, represented as a greyscale image. $\mathcal{K}$ filter kernels (in this example of size 3×3) are applied to the input, producing one pixel of the output activation map per possible filter position in the input. In this example, the kernels have been selected to perform different types of edge detection—in a real convolutional layer, the kernels are learned from data and not specified manually. The biases have been omitted to make the illustration more legible.

Intuitively, the operation performed by a convolutional layer can be viewed as scanning the input data with a set of $\mathcal{K}$ learnable filters. Each filter can learn to detect a particular pattern or feature in the local regions of the input data, such as edges, corners, or textures. By sliding the filter across the entire input space and computing the dot product between the filter's weights and the input region's values, the convolutional layer creates an activation map that highlights where the filter's pattern is found in the input data. This process is repeated for each kernel in the layer, generating

multiple activation maps that capture various features from the input. The use of multiple kernels allows the layer to learn a diverse set of features, which can be combined in subsequent layers to create higher-level representations.

From a more technical perspective, the convolutional layer can be viewed as a subtype[8] of the fully connected layer with the following modifications:

> **Sparse connectivity:** Each output neuron is connected to a local region of the input instead of all input neurons, reducing the number of weights to be learned.
>
> **Weight sharing:** The same set of weights (i.e., the kernel) is applied to all local input regions, reducing the number of parameters further.

Together, these modifications cause convolutional layers to be *translation invariant*: shifting the input $\boldsymbol{x}$ (horizontally and/or vertically) results in the output shifting identically[85].This is an important property in some image classification tasks, where the location of a detected pattern (e.g. a face) should not impact the decision of the classifier.

Pooling Layers

Pooling layers[172, 173] are a common component in contemporary CNNs, introduced to reduce the spatial dimensions of the output activation maps produced by convolutional layers. By downsampling the feature maps, pooling layers can help control model complexity, reduce computational requirements, and arguably improve the network's ability to generalize by introducing a form of spatial invariance. The most common types of pooling operations are max pooling and average pooling; max pooling selects the maximum value from a local region of the input feature map, whereas average pooling computes the average value of the same region[174].

Pooling layers can be seen as summarizing the presence of certain features in the input activation maps. In the case of max pooling, the layer retains the most prominent feature in the pooling region, while average pooling provides a smoother representation of the feature's presence.

While pooling layers have been successfully incorporated in CNN architectures for a plethora of applications, they have faced criticism for potentially discarding valuable spatial information and thus bounding performance below the theoretical maximum[175]. Many alternative approaches, such as strided convolutions[176] and global pooling[177], have been explored to maintain spatial information while still reducing activation map dimensions.

4.3.2 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a class of neural networks designed to process sequential data, capturing dependencies across time steps[178]. Unlike feedforward networks, RNNs maintain a hidden state that can store information from previous time steps, allowing the network to learn and utilize temporal patterns in the input data.

Given an input sequence $\boldsymbol{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_t, ..., \boldsymbol{x}_T\}$, where $\boldsymbol{x}_t \in \mathbb{R}^d$, a basic RNN layer updates its hidden state $\boldsymbol{h}_t \in \mathbb{R}^h$ and computes its output $\boldsymbol{o}_t \in \mathbb{R}^q$ at each time step t using the following equations:

$$\begin{aligned} \boldsymbol{h}_t &= \phi_h(\boldsymbol{W}^{hh}\boldsymbol{h}_{t-1} + \boldsymbol{W}^{xh}\boldsymbol{x}_t + \boldsymbol{b}_h), \\ \boldsymbol{o}_t &= \phi_o(\boldsymbol{W}^{ho}\boldsymbol{h}_t + \boldsymbol{b}_o), \end{aligned} \qquad \text{(eq. 4.22)}$$

where ϕ^h and ϕ^o are activation functions, $\boldsymbol{W}^{hh} \in \mathbb{R}^{h\times h}$, $\boldsymbol{W}^{xh} \in \mathbb{R}^{h\times d}$, and $\boldsymbol{W}^{ho} \in \mathbb{R}^{q\times h}$ are weight matrices, and $\boldsymbol{b}^h \in \mathbb{R}^h$ and $\boldsymbol{b}^o \in \mathbb{R}^q$ are bias vectors. The hidden state $\boldsymbol{h}_t$ (with size h, a hyperparameter) acts as a memory, allowing the RNN to capture information from previous time steps and use it to influence its current predictions.

To create a deep recurrent neural network, $L > 1$ RNN layers can be stacked on top of each other, where the hidden state output from the ith layer serves as the input for the subsequent layer, i.e. $\boldsymbol{h}_t^i = \boldsymbol{x}_t^{i+1}$. A final output can then be set to $\hat{\boldsymbol{y}}_t = \boldsymbol{o}_t^L$.

Long Short-Term Memory (LSTM)

While RNNs have shown success in learning short-term dependencies, they often struggle with capturing long-range dependencies due to issues such as vanishing or exploding gradients during training[179]. Long Short-Term Memory (LSTM) layers[180], a modified type of recurrent layer, were introduced to ameliorate this issue.

LSTMs introduce a more sophisticated memory cell structure that can maintain and manipulate information over longer sequences, in theory allowing them to capture long-range dependencies more effectively.

The LSTM cell consists of an input gate $\boldsymbol{i}_t$, a forget gate $\boldsymbol{f}_t$, an output gate $\boldsymbol{o}_t$, and a cell state $\boldsymbol{c}_t$. At each time step, the LSTM cell updates its internal state using:

$$
\begin{aligned}
\boldsymbol{i}_t &= \sigma(\boldsymbol{W}^{xi}\boldsymbol{x}_t + \boldsymbol{W}^{hi}\boldsymbol{h}_{t-1} + \boldsymbol{b}^i),\\
\boldsymbol{f}_t &= \sigma(\boldsymbol{W}^{xf}\boldsymbol{x}_t + \boldsymbol{W}^{hf}\boldsymbol{h}_{t-1} + \boldsymbol{b}^f),\\
\boldsymbol{o}_t &= \sigma(\boldsymbol{W}^{xo}\boldsymbol{x}_t + \boldsymbol{W}^{ho}\boldsymbol{h}_{t-1} + \boldsymbol{b}^o),\\
\tilde{\boldsymbol{c}}_t &= \tanh(\boldsymbol{W}^{xc}\boldsymbol{x}_t + \boldsymbol{W}^{hc}\boldsymbol{h}_{t-1} + \boldsymbol{b}^c),\\
\boldsymbol{c}_t &= \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t,\\
\boldsymbol{h}_t &= \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t), \qquad \text{(eq. 4.23)}
\end{aligned}
$$

where σ is the sigmoid activation function from eq. 4.2, and $\boldsymbol{W}^{uv}$ and $\boldsymbol{b}^v$ ($u, v \in \{x, i, f, o, c\}$) are weight matrices and bias vectors, respectively. The input gate $\boldsymbol{i}_t$ controls the degree to which new input information is incorporated into the cell state, the forget gate $\boldsymbol{f}_t$ regulates how much of the previous cell state is retained, and the output gate $\boldsymbol{o}_t$ determines the contribution of the cell state to the hidden state.

A popular modification of the LSTM architecture is the introduction of *peephole connections*, which allow the gates $\boldsymbol{i}_t$, $\boldsymbol{f}_t$, and $\boldsymbol{o}_t$ to not only depend on the previous hidden state $\boldsymbol{h}_{t-1}$, but also the cell state $\boldsymbol{c}_{t-1}$, improving the model's ability to handle precise timing problems[181].

Gated Recurrent Unit (GRU)

The Gated Recurrent Unit (GRU)[182] is another variant of RNN that was introduced as a more computationally efficient alternative to LSTMs while still addressing the vanishing and exploding gradient problem. Similar to LSTMs, GRUs employ gating mechanisms to control the flow of information through the network, enabling them to capture long-range dependencies in sequential data. However, GRUs simplify the LSTM architecture by using fewer gates and combining the hidden state and cell state into a single state.

A GRU cell consists of an update gate $\boldsymbol{z}_t$ and a reset gate $\boldsymbol{r}_t$. At each time step, the GRU cell updates its hidden state $\boldsymbol{h}_t$:

$$
\begin{aligned}
\boldsymbol{z}_t &= \sigma(\boldsymbol{W}^{xz}\boldsymbol{x}_t + \boldsymbol{W}^{hz}\boldsymbol{h}_{t-1} + \boldsymbol{b}^z), && \text{(eq. 4.24)}\\
\boldsymbol{r}_t &= \sigma(\boldsymbol{W}^{xr}\boldsymbol{x}_t + \boldsymbol{W}^{hr}\boldsymbol{h}_{t-1} + \boldsymbol{b}_r), && \text{(eq. 4.25)}\\
\tilde{\boldsymbol{h}}_t &= \tanh(\boldsymbol{W}^{xh}\boldsymbol{x}_t + \boldsymbol{W}^{hh}(\boldsymbol{r}_t \odot \boldsymbol{h}t - 1) + \boldsymbol{b}_h), && \text{(eq. 4.26)}\\
\boldsymbol{h}_t &= (1 - \boldsymbol{z}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{z}_t \odot \tilde{\boldsymbol{h}}_t, && \text{(eq. 4.27)}
\end{aligned}
$$

The update gate $\boldsymbol{z}_t$ controls the degree to which the previous hidden state is maintained, while the reset gate $\boldsymbol{r}_t$ modulates the incorporation of the previous hidden state when computing the candidate hidden state $\tilde{\boldsymbol{h}}_t$.

GRUs have been shown to exhibit competitive performance with LSTMs on many sequence processing tasks while being less computationally expensive (both w.r.t. time and memory) for a given size of the hidden state[183].

4.3.3 Transformers

Transformers[184] are a type of ANN model architecture that maps an input sequence $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n) \in \mathbb{R}^{n \times \mathcal{E}_m}$, where $\boldsymbol{x}_i \in \mathbb{R}^{\mathcal{E}}$ is an input feature vector, to an output sequence $\hat{\boldsymbol{Y}} = (\hat{\boldsymbol{y}}_1, \hat{\boldsymbol{y}}_2, ..., \hat{\boldsymbol{y}}_m) \in \mathbb{R}^{m \times \mathcal{E}_m}$, where $\hat{\boldsymbol{y}}_j \in \mathbb{R}^{\mathcal{E}_m}$. The embedding size $\mathcal{E}_m \in \mathbb{N}$ is a hyperparameter of the model—to process sequences of input feature vectors with dimension $d \neq \mathcal{E}_m$, they most first be remapped. Similarly, for the model to produce outputs of dimension $l \neq \mathcal{E}_m$, the elements of the output sequence $\hat{\boldsymbol{Y}}$ should be postprocessed (e.g. linearly projected) appropriately.

The transformer comprises two components, an encoder and a decoder. The encoder maps the input sequence $\boldsymbol{X}$ to a continuous representation $\boldsymbol{R} \in \mathbb{R}^{n \times \mathcal{E}_m}$. The decoder generates the output sequence $\boldsymbol{Y}$ by processing $\boldsymbol{R}$.

Both the encoder and the decoder are composed of L identical layers, each of which operates on the output sequence of the previous layer. Each layer of both the encoder and the decoder is composed of two modules: a multi-head attention module followed by an element-wise MLP module.

The multi-head self-attention module computes an output sequence whose elements are constructed as a nonlinear combinations of input sequence elements:

$$\begin{aligned} &\boldsymbol{Z} = \text{MultiHead}(\boldsymbol{I}; \boldsymbol{W}^Q, \boldsymbol{W}^K, \boldsymbol{W}^V, \boldsymbol{W}^O) = \text{Concat}(\boldsymbol{H}^1, ..., \boldsymbol{H}^h) \cdot \boldsymbol{W}^O, \\ &\text{where } \boldsymbol{H}^i = \text{Attention}(\boldsymbol{I}; \boldsymbol{W}^Q_{i,*,*}, \boldsymbol{W}^K_{i,*,*}, \boldsymbol{W}^V_{i,*,*}) \end{aligned} \qquad \text{(eq. 4.28)}$$

Where $\boldsymbol{I} \in \mathbb{R}^{n \times \mathcal{E}_m}$ is the input sequence and $\boldsymbol{Z} \in \mathbb{R}^{n \times \mathcal{E}_m}$ is the output sequence of the module.

In the encoder, the attention mechanism in eq. 4.28 can be expressed as:

$$\text{Attention}(\boldsymbol{I}; \boldsymbol{W}^q, \boldsymbol{W}^k, \boldsymbol{W}^v) = \text{softmax}\left(\frac{\boldsymbol{Q} \cdot \boldsymbol{K}^T}{\sqrt{\mathcal{E}_k}}\right) \cdot \boldsymbol{V}$$
$$\text{where } \boldsymbol{Q} = \boldsymbol{I}\boldsymbol{W}^q \,,\, \boldsymbol{K} = \boldsymbol{I}\boldsymbol{W}^k \,,\, \boldsymbol{V} = \boldsymbol{I}\boldsymbol{W}^v \qquad \text{(eq. 4.29)}$$

Here, the softmax operation is performed row-wise. The matrices $\boldsymbol{Q} \in \mathbb{R}^{n \times \mathcal{E}_k}$, $\boldsymbol{K} \in \mathbb{R}^{n \times \mathcal{E}_k}$, and $\boldsymbol{V} \in \mathbb{R}^{n \times \mathcal{E}_v}$ are called the *queries*, *keys*, and *values*, respectively. This type of attention, where these matrices are calculated solely from $\boldsymbol{I}$, is called *self* -attention

In the decoder, the attention mechanism in eq. 4.28 takes an additional input $\boldsymbol{R}$, which as previously mentioned is the output of the last encoder layer. $\boldsymbol{R}$ is used to compute the key matrix and value matrix, whereas the input $\boldsymbol{I}$ from the previous decoder layer is used to compute the query matrix:

$$\text{Attention}(\boldsymbol{I}, \boldsymbol{R}; \boldsymbol{W}^q, \boldsymbol{W}^k, \boldsymbol{W}^v) = \text{softmax}\left(\frac{\boldsymbol{Q} \cdot \boldsymbol{K}^T}{\sqrt{\mathcal{E}_k}}\right) \cdot \boldsymbol{V}$$
$$\text{where } \boldsymbol{Q} = \boldsymbol{I}\boldsymbol{W}^q \,,\, \boldsymbol{K} = \boldsymbol{R}\boldsymbol{W}^k \,,\, \boldsymbol{V} = \boldsymbol{R}\boldsymbol{W}^v \qquad \text{(eq. 4.30)}$$

In terms of parametrization, $\boldsymbol{W}^Q \in \mathbb{R}^{h \times \mathcal{E}_m \times \mathcal{E}_k}$, $\boldsymbol{W}^K \in \mathbb{R}^{h \times \mathcal{E}_m \times \mathcal{E}_k}$, and $\boldsymbol{W}_i^V \in \mathbb{R}^{h \times \mathcal{E}_m \times \mathcal{E}_v}$ are learnable weight tensors, and $\boldsymbol{W}^O \in \mathbb{R}^{h\mathcal{E}_v \times \mathcal{E}_m}$ is an output weight matrix that governs how the output sequence is recombined from the outputs of the heads. The number of heads h is a hyperparameter, as is the key embedding size $\mathcal{E}_k$ and the value embedding size $\mathcal{E}_v$.

In the element-wise MLP module, a fully connected feed-forward network is applied independently to each position in the sequence to produce the final output sequence $\boldsymbol{O} \in \mathbb{R}^{n \times \mathcal{E}_m}$ of the layer. This subnetwork is typically a two-layer MLP (defined in section 4.1) with a ReLU activation function for the hidden layer and a linear activation function for the output layer:

$$\boldsymbol{O}_{i,*} = \text{MLP}(\boldsymbol{Z}_{i,*}; \boldsymbol{W}^1, \boldsymbol{b}^1, \boldsymbol{W}^2, \boldsymbol{b}^2) \qquad \text{(eq. 4.31)}$$

In addition to the mappings described in eq. 4.28 and eq. 4.31, both the multi-head attention module and the MLP module make use of *layer normalization*[185] and *residual connections*[136]. These techniques are described in section 4.5.

The first layer of the encoder operates directly on the input $\boldsymbol{X}$. The first layer of the decoder, on the other hand, uses the continuous representation $\boldsymbol{R}$ generated by the last

layer of the encoder as well as an additional input sequence $\boldsymbol{Y}'$ that typically corresponds to the partially generated output. In autoregressive modelling, $\boldsymbol{Y}'$ is typically a shifted version of the target sequence during training or the output generated so far during inference. To prevent the decoder from accessing future information in the target sequence, a masking technique is applied to the self-attention mechanism, ensuring that the decoder can only attend to positions up to and including the current one. This masking is typically implemented by setting the attention weights of the future positions to a very large negative value before applying the softmax function.

In the increasingly popular but confusingly named[9] *encoder-only* transformer architecture[97, 186], the model is not separated into an encoder and a decoder. Instead, a single component comprised of a stack of identical transformer layers that makes use of the attention formulation from eq. 4.29 is used to map the input sequence $\boldsymbol{X}$ directly to the output sequence $\hat{\boldsymbol{Y}}$.

As the self-attention mechanism in the Transformer architecture is inherently permutation-invariant, it lacks the ability to capture the positional information of the elements in the input sequence. To address this issue, positional encodings is introduced to provide the model with information about the relative positions of the input elements. There are two common approaches to positional encoding: learned positional encoding and sinusoidal positional encoding.

In learned positional encoding[187], the model is equipped with a learnable positional embedding matrix $\boldsymbol{P} \in \mathbb{R}^{n \times \mathcal{E}_m}$ which is initialized randomly and optimized during training. Each row of $\boldsymbol{P}$ represents the positional encoding for a specific position in the sequence. The positional embeddings are added to the input sequence element-wise before being fed into the Transformer, i.e. $\boldsymbol{I}^1 = \boldsymbol{X} + \boldsymbol{P}$. This approach allows the model to learn the optimal representation of position information for the given task.

In sinusoidal positional encoding[184], the position information is encoded using a predefined fixed function based on sine and cosine waves. The sinusoidal positional encoding is computed for each position in the sequence and then added element-wise to the input sequence. The main advantage of the sinusoidal positional encoding is that it can generalize to input sequences of varying lengths, as the function can be computed for any position index. Furthermore, it does not require any additional learnable parameters, which can be beneficial for memory efficiency and reducing the risk of overfitting.

Since its inception, the transformer architecture has been widely applied to a variety of tasks (natural language processing[188], vision[186], speech[189], etc.) with almost uniformly impressive results. Even so, a significant drawback of the transformer architecture compared to other sequence processing models (such as RNNs) is the

computational requirements of attention. Attention of the form presented in eq. 4.29 requires comparing (the query $\boldsymbol{Q}_{i,*}$ of) every element of the input sequence with (the key $\boldsymbol{K}_{j,*}$ of) every other element, leading to quadratic growth (w.r.t. sequence length n) of both time and memory complexity. Several modifications of the standard transformer have been proposed to eliminate or reduce this drawback: Performer[190], Longformer[191], Perceiver[192], among many others[193].

4.4 Regularization

As was discussed in section 3.4, machine learning models with excessive capacity for representing functions runs risk of overfitting to limited training data. *Regularization* denotes techniques for preventing overfitting and improving the generalization performance of models, for example by limiting capacity or reducing reliance on individual training examples. Some commonly employed countermeasures to overfitting (in the work of this dissertation and elsewhere) in ANN models are briefly described in this section.

Weight Penalties

L1 and L2 regularization (sometimes referred to as *weight decay*) involve adding a penalty term to the loss function, which discourages large weights in the model. In L1 regularization, the penalty term is the sum of the absolute values of the weights, while in L2 regularization, it is the sum of the squared values of the weights[84].

Given a loss function $\mathcal{L}(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta})$ of some supervised learning task (examples of which are presented in section 4.2), where $\boldsymbol{\theta} = \{\{\boldsymbol{W}_1, \boldsymbol{b}_1\}, \{\boldsymbol{W}_2, \boldsymbol{b}_2\}, ...\}$, the regularized loss functions for L1 and L2 regularization can be written as:

$$\mathcal{L}^{L1}(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta}) + \lambda \sum_{\boldsymbol{W} \in \boldsymbol{\theta}} \sum_{w \in \boldsymbol{W}} |w| \qquad \text{(eq. 4.32)}$$

$$\mathcal{L}^{L2}(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta}) + \frac{\lambda}{2} \sum_{\boldsymbol{W} \in \boldsymbol{\theta}} \sum_{w \in \boldsymbol{W}} w^2, \qquad \text{(eq. 4.33)}$$

where $\lambda \geq 0$ is a hyperparameter that determines the strength of the penalty term. The main difference between L1 and L2 regularization is that L1 regularization incentivizes sparsity in the learned weights—in some problems leading to a more interpretable and potentially more robust model—while L2 regularization tends to spread the weights more evenly.

The mechanism(s) by which L1 and L2 regularization prevent overfitting are not entirely understood, but can informally (and perhaps wrongly, cf. section 4.6) be viewed through the lens of model capacity. Loosely speaking, weight penalties biases training towards parameter sets with smaller magnitudes, resulting in models whose mappings from input to output contain fewer rapid changes and less (high-dimensional equivalents of) steep slopes. In a sense, this is related to the broad preference given to simpler hypothesis and solutions in many truth-seeking pursuits (cf. *Occam's Razor*[194]). While there is no fundamental reason to expect this to be the case *a priori* for all conceivable problems, it seems to be a robust fact about the world and the data-generating distributions it contains that simpler (in the Kolmogorov complexity[195] sense, for example) models are overrepresented[196] (see the somewhat related discussion in section 3.4).

Dropout

Dropout[197] is a simple yet surprisingly effective regularization technique ubiquitous in contemporary ANN training. During the training phase, the method works by setting a fraction of the neuron outputs to zero at each training iteration. The probability p of a neuron being dropped out is a hyperparameter set to a value between 0 and 1. p can in principle be set individually for every neuron of a network, but is in practice usually specified either globally or layer-wise.

Dropout can be more formally defined as follows. Let $\boldsymbol{a}$ denote the vector of activations of a hidden layer, and $\boldsymbol{a'}$ the corresponding activations after applying dropout. Then, the dropout operation can be described as:

$$a'_i = a_i \cdot (1 - \mathcal{B}(p)), \qquad \text{(eq. 4.34)}$$

where $\mathcal{B}(p)$ is a Bernoulli random variable with probability p of being 1, and i indexes the hidden units. During inference, the dropout operation is replaced by:

$$a'_i = a_i \cdot \frac{1}{p} \qquad \text{(eq. 4.35)}$$

Dropout introduces noise and uncertainty into the learning process by randomly setting a fraction of neurons' activations to zero during training. Informally, this forces the model to learn *redundant* representations, as it cannot rely on the presence of any single neuron. This redundancy improves the model's generalization ability, as it becomes more robust to small perturbations in the input data.

Relatedly, dropout has since its inception been interpreted as a type of *Baysesian learning*[198]. In this view, training a network with dropout is mathematically equivalent to training an ensemble of neural networks with shared weights, where each network is obtained by sampling from the original network's architecture.

Label Smoothing

Label smoothing is another straightforward approach aimed at reducing model overconfidence. Under label smoothing, models training proceeds as usual, but one-hot ground truth target vectors $\boldsymbol{Y} = \{\boldsymbol{y}^1, ..., \boldsymbol{y}^n\}$, where $\boldsymbol{y}^i \in \{0, 1\}^c$, are smoothed before being fed to the loss function:

$$\tilde{y}_j^i = (1 - \alpha) y_j^i + \frac{\alpha}{c}, \qquad \text{(eq. 4.36)}$$

This encourages the model to be less certain about its predictions and can improve generalization.

Early Stopping

Early stopping is a conceptually simple regularization technique that monitors the model's performance on a held out *validation set* during training. Crucially, the validation set is never used to update the model during training, and is typically constructed by simply subdividing all available training examples into two disjoint sets, one which is used for iteratively fitting the model as described in section 4.2 and the other for validation. In practice, this partitioning can either be performed randomly or via some predetermined rule appropriate for the data-generating process. The training process is halted when the performance on the validation set starts to degrade, indicating that the model is overfitting the training data. By monitoring the model's performance on a validation set and halting the training when the performance starts to degrade, early stopping prevents the model from learning noise or other confounding patterns present in the training set. This approach in effect estimates an optimal stopping point for the training process, ensuring that the model does not overfit the training data and achieves better generalization.

Data Augmentation

Data augmentation refers to methods for increasing the size and diversity of a training dataset for artificial neural networks (ANNs) by applying various transformations to

the original examples. In vision problems, such transformations can include rotation, scaling, and flipping[199]. In signal processing, copying and corrupting the dataset with varying levels of (typically Gaussian) noise is common[200].

By creating more diverse training data, data augmentation can cause the model to learn features that are less likely to rely on spurious axes of input data variation.

4.5 Auxiliary Techniques

The fundamental steps necessary for constructing a DL model is architecture design, loss formulation and minimization, and regularization. In addition, a number of augmentations have been introduced and reached widespread use in the preceding decade. A limited selection of techniques that have been found to aid in convergence and generalization are described in this section.

Normalization

Normalization techniques help to mitigate the problem of internal covariate shift, which occurs when the distribution of inputs to a layer changes during training[10]. This can lead to slower convergence and make the network highly sensitive to the choice of hyperparameters[201]. Multiple normalization techniques have been proposed to address this issue.

Batch normalization[202] normalize input examples with respect to summary statistics computed batch-wise during training. Given a training batch $\mathcal{B} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_m\} \subseteq \boldsymbol{X}$, batch normalization computes the sample (feature-wise) mean $\mu_{\mathcal{B}}$ and variance $\boldsymbol{\sigma}^2_{\mathcal{B}}$:

$$\mu_{\mathcal{B}} = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{x}_i$$
$$\boldsymbol{\sigma}_{\mathcal{B}}\hat{}2 = \frac{1}{m}\sum_{i=1}^{m}(\boldsymbol{x_i} - \mu_{\mathcal{B}})\hat{}2 \qquad \text{(eq. 4.37)}$$

Then, the normalized feature values $\hat{x}_i$ are calculated as:

$$\hat{\boldsymbol{x}}_i = \frac{\boldsymbol{x}_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{(eq. 4.38)}$$

where ϵ is a small constant added for numerical stability. Finally, the batch normalization layer applies a learnable affine transformation to the normalized features:

$$y_i = \boldsymbol{\gamma}\hat{\boldsymbol{x}}_i + \boldsymbol{\beta} \tag{eq. 4.39}$$

where γ and β are learnable parameters with the same shape as the input features.

Layer normalization[185] is an alternative to batch normalization that does not depend on sufficiently large batches to compute summary statistics from. Unlike batch normalization, layer normalization computes the mean and variance across the features of each individual input example in the layer. Given a layer input $\boldsymbol{x} \in \mathbb{R}^d$, layer normalization computes the mean μ_i and variance σ_i^2 as:

$$\begin{aligned}\mu &= \frac{1}{d}\sum_{i=1}^{d} x_i \\ \sigma^2 &= \frac{1}{d}\sum_{i=1}^{d}(x_i - \mu_i)^2\end{aligned} \tag{eq. 4.40}$$

The normalization and affine transformation steps are identical to those in batch normalization:

$$\hat{x}\boldsymbol{i} = \frac{x_i - \mu_i}{\sqrt{\sigma_i\hat{}\,2 + \epsilon}} \tag{eq. 4.41}$$

$$y_i = \gamma_i\hat{x}_i + \beta_i \tag{eq. 4.42}$$

where $\boldsymbol{\gamma} \in \mathbb{R}^d$ and $\boldsymbol{\beta} \in \mathbb{R}^d$ are learnable parameters for each feature.

Residual Connections

An important innovation that has enabled the training of truly deep ($L > 100$) CNN and transformer models is the introduction of *residual connections*, or, synonymously, *skip connections*[136]. Given a standard feedforward layer $f^{(l)}$, its output is computed as $\boldsymbol{a} = f(\boldsymbol{x};\boldsymbol{\theta})$. Using a residual connection, the layer is modified to instead perform the mapping:

$$\boldsymbol{a}^l = f^{(l)}(\boldsymbol{x};\boldsymbol{\theta}) + \boldsymbol{x} \tag{eq. 4.43}$$

In words, the modification entails simply adding the input to the output of the layer before feeding it to the next layer.

An intuition behind the advantage of residual connections is that they facilitate the learning of identity mappings. In other words, the network can learn to output the same value as the input, which is useful when the optimal function is close to the identity function.

4.6 Conjectures & Conundrums

DL systems has achieved empirical success across a wide range of applications, outperforming traditional algorithms and occasionally even human-level benchmarks[203]. However, a successful mechanistic or functional theory of statistical learning as it relates to such systems has yet to emerge[204]. This section reviews some recent observations and corresponding hypotheses that aim to explain why and how DL systems perform at a remarkably high (compared to other learning frameworks) level for a variety of tasks.

The Importance of Depth

Model depth (i.e. large numbers of consecutive layers) has for a long time been suggested as playing an important role in the success of DL systems, as it intuitively[11] should allow models to learn efficient hierarchical feature representations[31]. Empirically, this view has been well-supported—deep architectures consistently outperform their shallow counterparts in various tasks, such as image classification [**?**] and natural language processing[184].

While scarce, formal theories for the benefits of depth include the expressive efficiency of deep networks[205, 206], which posits that deep models can represent certain functions more compactly (i.e. with fewer parameters) than shallow models, and the hierarchical representation of features, wherein lower layers has been shown learn simple, local features and higher layers can capture more abstract, global information[207].

The Unreasonable Effectiveness of Gradient Descent

At face value, the success of SGD (and its adaptive variants) as the algorithm of choice for fitting ANN models to large datasets is somewhat surprising [208]. Despite being a local, first-order optimization method (and fitting ANNs to large datasets typically being

highly nonconvex problems with multiple local minima[209]), SGD has repeatedly been shown to yield solutions with low error rate, often outperforming more complicated optimization schemes. More confusing yet, solutions found by SGD frequently seem to generalize well to unseen data[135].

One class of explanations for the effectiveness of SGD emphasizes its implicitly regularizing properties. In particular, the noise introduced by using randomly sampled small batches during optimization has been shown to guiding the algorithm towards solutions with better generalization performance than large-batch training[210]. Additionally, the convergence to wider minima in the loss landscape, which typically (for poorly under stood reasons) corresponds to network configurations with better generalization, has been observed for SGD [211].

It has recently been noted[212] that the classic view of generalization has very weak empirical support in overparametrized DL systems, where the number of model parameters exceeds the size of the training data. Loosely speaking, the bias-variance trade-off (discussed in section 3.4) implies that out-of-sample performance is expected to decrease with excess model capacity in noisy real-world problems. This is in sharp contrast to the empirically supported *double descent phenomenon*[213, 214] (illustrated in Fig. 4.3). As the size of an ANN model increases, test error initially decreases as the model capacity increases, then increases into the classical overfitting regime as expected. However, as the parameter count grows further, the test error starts decreasing again and eventually falls below the previous optimum. This phenomenon is arguably related to the *lottery ticket hypothesis*, which posits that large networks contain sparse subnetworks (so-called 'winning tickets') that, when trained in isolation, can achieve comparable performance to the full network[215]. This hypothesis suggests that the success of extremely overparametrized networks are due to the presence of a larger number of 'winning tickets' following parameter initialization[216].

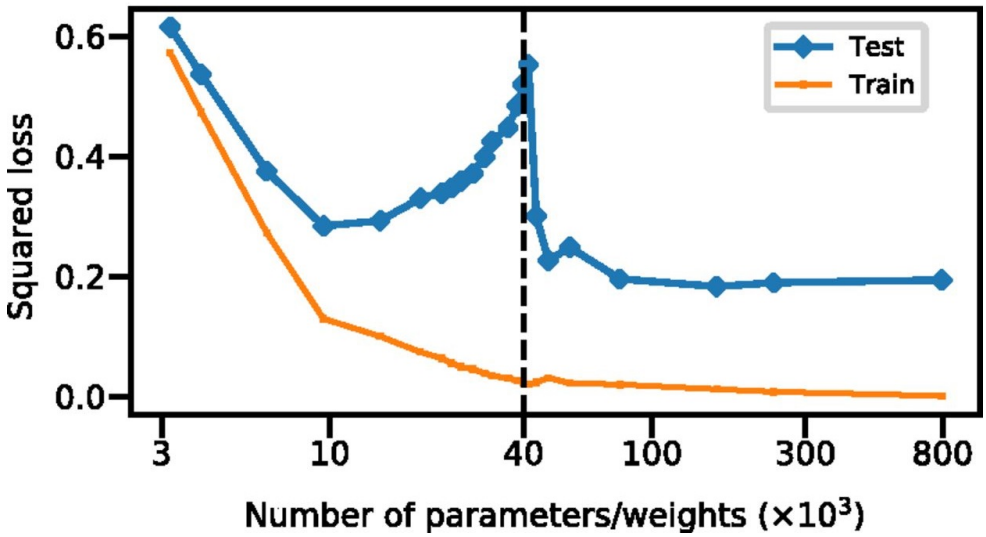


Figure 4.3: The double descent phenomenon. On many supervised ML tasks, overfitting does not seem to occur for sufficiently overparametrized ANN models. Figure modified from Belkin *et al.* [213], ©PNAS.

Great ideas originate in the muscles.

Thomas A. Edison

Chapter 5

Muscle-Computer Interfacing

Seamless integration of man and machine has long been a subject of fascination in both art and reality[217]. Although science fiction[12] often envisions interfacing occurring at the level of the CNS, a far more practical point of neural interception is at the level of skeletal muscles[19]. As is covered in chapter 2, volitional muscle contractions are accompanied by local electrical activity that can be transduced via EMG. The availability of EMG signals forms the principal motive for pursuing Muscle-Computer Interfaces (MCIs)[218]—systems that transform myoelectricity from a mere correlate of muscle activity to actionable output commands.

MCIs have found use in a number applications and been identified as a promising research avenue in many others[218]. In prosthetics (see chapter 6), MCIs can be used to facilitate control of artificial limbs [219, 26, 27]. In rehabilitation, MCIs can be used to restore motor functions in patients with neuromuscular impairments, such as stroke, through for example augmenting motor relearning[220] or informing the behaviour of an assistive exoskeleton[221]. Outside the realm of the clinical, MCIs have seen tentative use in virtual reality and gaming, where they could serve as a control interface for users to navigate virtual environments or manipulate game elements[20, 222]. Common to almost all aforementioned applications is the need for mappings between EMG signals and relevant representations of concurrent motor intent (limb kinematics[223], encoded grasps[224], latent distributions of motor unit firings[225], *etc.*). This signal processing task, which is the topic of this chapter, is commonly and appropriately referred to as *motor intent decoding*.

Section 5.1 defines some commonly reported properties that make motion-decoding MCI systems appealing. Section 5.2 presents the standard tools for EMG signal acquisition and preprocessing for the purpose of facilitating subsequent extraction of intent. Section 5.3 reviews algorithms for mapping preprocessed signals to output motion commands. Section 5.4 gives examples for evaluative frameworks that aim to quantify motor decoding performance. Lastly, section 5.5 briefly lists and comments on some alternative, non-EMG modalities that have been proposed for use with MCIs.

5.1 Desiderata

Robustness

In the context of MCI, robustness refers to the ability of a system to maintain consistent and reliable performance in the presence of noise and shifts in EMG signal properties (brought about by, for example, fatigue, changes in body position, and perspiration-induced changes in electrode-skin impedance)[226, 29]. As such, the robustness of an MCI system influences the overall reliability of the system across operating conditions.

Naturalness

Naturalness of control refers to the degree to which an MCI system allows users to perform movements in a manner that resembles innate motor behaviour[28]. This is particularly relevant in the context of prosthetics, where the goal is to replicate physiological movements to facilitate activities of daily living. The main promise of *pattern recognition control* (discussed in section 5.3) is to provide high naturalness by mapping EMG patterns to output commands corresponding directly to the moment performed by the user[227, 228].

Simultaneity of Control

Simultaneous control refers to the capacity of an MCI system to manage multiple degrees of freedom (DoFs) concurrently, enabling the user to execute coordinated movements involving multiple joints[229, 230]. While arguably not strictly necessary for all applications (e.g. a prosthesis might provide sufficient function even if only able to perform a preselected set of grasps), simultaneous control is clearly an inherent property of natural motor function in healthy individuals.

Proportionality of control

Proportionality of control refers to the ability of an MCI system to modulate the magnitude of the generated output command in relation to the intensity of the user's muscle activity[231]. This is relevant in scenarios where the user needs to perform movements with varying force or speed, as it allows for a graded control of the output based on the user's intent[225]. Proportionality can be seen as a property that may contribute to the overall naturalness and functionality of an MCI system, depending on the specific requirements of the intended application.

5.2 EMG Acquisition & Processing

By definition, all interfaces for myoelectric decoding functions by acquiring and processing EMG signals. An archetypical processing pipeline from raw EMG to decoded motion intent is illustrated graphically in Fig 5.1.

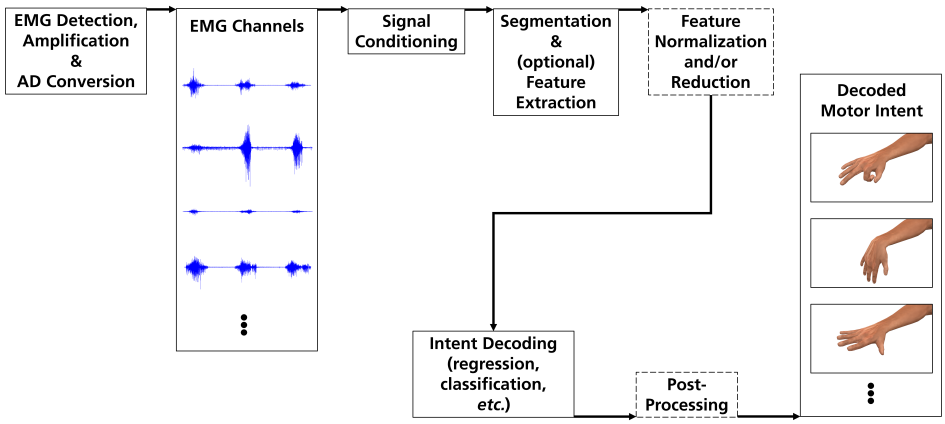


Figure 5.1: Typical signal processing pipeline from EMG signals to decoded motor intent.

Sampling

Due to the low amplitude of raw iEMG and sEMG signals (typically in the range of 1-2 mV during voluntary contractions), amplification (with gain of around 60 dB) with is usually required before further processing[232]. Differential amplifiers with high common-mode rejection ($\gtrapprox$ 100 dB) are commonly required in EMG systems to reduce common-mode noise (e.g., interference from power lines and other electronic devices) which can otherwise be of prohibitively high amplitude in relative terms[233]. As the energy content of surface EMG is largely concentrated to frequencies between 10 and 500 Hz, analog-to-digital sampling rates between 1 and 10 kHz (as dictated by the Shannon-Nyquist sampling theorem[234]) are typically used for sEMG[235]. For iEMG, which unlike sEMG is not naturally low-pass filtered by the capacitance of tissue, may (depending on application) require higher sampling rates[236].

Preprocessing

Following the acquisition of digital EMG signals, preprocessing techniques are often applied to condition the data and mitigate the influence of noise or artifacts that might compromise subsequent analysis. *Baseline correction* involves removing any DC offset from the signal, ensuring that the EMG waveform is centered around zero[237]. Filtering techniques, such as bandpass filtering, are commonly employed to extract the frequency components of interest (typically in the vicinity of 20-500 Hz) and remove unwanted noise such as power line interference and motion artifacts[225]. *Outlier removal* is another set of preprocessing steps that aims to identify and eliminate data points or segments that are inconsistent with the expected characteristics of the EMG signals, possibly due to movement artifacts, noise, or other external factors[238]. A simple such method is the use of *clipping*, where individual voltage samples outside some range of appropriate values are assumed to not be genuine reflections of muscle activity and are as such rounded down[239].

Segmentation & Feature Extraction

After relevant preprocessing, *segmentation* it typically used to transform the continuous EMG waveform into separate windows that can be processed end-to-end in subsequent motor decoding[27]. The standard approach to segmentation is the sliding window technique, wherein a fixed-length window is moved across the EMG signal in discrete increments, referred to as the window step size. The selection of an appropriate window width and step size has significant bearing on subsequent analysis—shorter window steps result in higher temporal resolution, enabling more frequent updates of outputs thus a more responsive system. Naturally, this comes with increased computational costs, as all subsequent window-wise operations must occur more frequently. Similarly, the width of the window bounds the amount of information available in subsequent analysis. While too short windows almost universally results in poor decoder performance, the maximum appropriate window size depends heavily on the subsequent analysis[240, 241].

Feature extraction is of central importance in the processing of segmented EMG signals for pattern recognition-based MCI systems. This step aims to condense the information content of the raw data contained in windows into a more compact representation that can be used for the subsequent decoding of movement intent[242]. Concretely, features are intended to be as *discriminative* as possible, i.e. assuming different values for different possible motor intents. The importance of informative features is reflected in the large number of features and feature sets that have been proposed in the literature[243, 244].

A more recent development is the (highly relevant to the work of this dissertation) class of methods that do not employ any feature engineering but instead use parametrized models to *learn* instrumentally useful representations of the raw voltages of EMG windows, typically by leveraging deep ANN models[245]. Such methods are reviewed in section 5.3.

The Hudgins feature set[25], sometimes known as the TD4 features (where TD denotes time-domain), is arguably most popular set of features used in motor intent decoding. It consists of four computationally light-weight features: Mean Absolute Value (MAV), Zero Crossings (ZC), Slope Sign Changes (SSC), and Waveform Length (WL) that are extracted channel-wise from windows of raw sEMG voltages. Due to their common use as a benchmark in the work of this dissertation and elsewhere, they are defined below.

The MAV is a simple feature that represents the average amplitude of the EMG signal:

$$MAV = \frac{1}{N}\sum_{i=1}^{N}|x_i| \qquad \text{(eq. 5.1)}$$

where N is the number of samples in the analysis window, and x_i is the i-th sample in the EMG signal.

ZC is the number of times the EMG signal crosses the zero amplitude level. It is a measure of the frequency content of the signal:

$$ZC = \sum_{i=1}^{N-1}|sgn(x_{i+1} - x_i) - sgn(x_i)| \qquad \text{(eq. 5.2)}$$

SSC is the number of times the slope of the EMG signal changes sign. This feature helps to succinctly summarize the presence or absence of high-frequency components in the EMG signal.

$$SSC = \sum_{i=2}^{N-1} H(|x_{i+1} - x_i| \cdot |x_{i-1} - x_i|) \qquad \text{(eq. 5.3)}$$

WL is the cumulative length of the EMG signal waveform over the analysis window. It can be viewed as a simple measure of the signal's complexity.

$$WL = \sum_{i=1}^{N-1}|x_{i+1} - x_i| \qquad \text{(eq. 5.4)}$$

5.3 Motor Intent Decoding

Gesture Recognition

Myoelectric *gesture recognition* here denotes methods for motor intent decoding that, given an input EMG window, return a discrete gesture or grasp selected from a finite set of prespecified alternatives[246]. The principal reason for this problem formulation is its strong amenability to the type of classification methods reviewed in section 3.3.1. Using this framework, a training dataset containing (multiple repetitions of) attempts at performing every gesture that the MCI should be able to detect—together with concurrently acquired EMG signals—must first be collected from the prospective user. After preprocessing, segmentation, and (optionally) feature extraction, this data can be used to train a ML model. A large variety of classification algorithms available from the wider ML literature have been applied to this end, including but not limited to SVMs (e.g. [247]), kNNs (e.g. [248]), Random Forests (e.g. [249]), Hidden Markov Models (e.g. [250]), and arguably most popularly, LDA (e.g. [251, 252, 253]).

A large number of studies on classificatory methods based on DL have been conducted in recent years. As mentioned in section 5.2, such methods typically eschew feature engineering by operating on raw EMG. In addition to avoiding information loss from potentially false beliefs on the part of researchers on what constitutes discriminatory properties of EMG signals (cf. *the bitter lesson* from section 3.1), deep ANNs can leverage data and computational resources at scale to learn arbitrarily complicated nonlinear mappings from small EMG variations to output motion commands.

Several systems that make use of CNNs for decoding motor intent from sEMG have been proposed. Atzori *et al.*[254] applied a shallow (4 layers) CNN to raw 12-channel EMG and found it to exhibit slightly lower accuracy than state-of-the-art classical methods using traditional feature engineering. Since then, multiple systems (e.g. [255, 256, 257, 245]) based on deeper CNNs trained on more data have succeeded in outperforming traditional feature-dependent approaches in terms of offline performance metrics. Paper II [258] of this dissertation introduced a meta-optimization framework for designing CNN architectures specifically for EMG processing.

Geng *et al.*[259] demonstrated the feasibility of extracting gestures from extremely short time windows of spatially dense data by letting a CNN operate on raw, instantaneous HD-sEMG. A CNN-based pipeline for processing HD-sEMG for the purpose of simultaneous control is proposed in paper I [260] of this dissertation.

RNNs of various types have, given their inherent suitability for tasks with strong

intertemporal structure, been applied to EMG in several studies (e.g. [261, 262, 263, 264, 265, 266]). The access to temporal context afforded by such architectures can improve decoding accuracy above that achieved by window-based methods by providing appropriate 'priors' for the classification decision. However, with limited training data, this can result a period of increased error at the onset of movement[262], as the memory cells of the network have not yet updated to absorb the new context.

Like in many problem domains where input data can naturally be construed as a sequence of elements, early use of transformer models have generated promising results for EMG gesture recognition. Zabihi *et al.*[267] proposed a transformer-based architecture termed TraHGR, and demonstrated competitive performance with traditional pattern recognition, CNNs, and RNNs on open data. Rahimian *et al.*[268] introduced TEMGNet, a vision (i.e. encoder-only) transformer for EMG gesture classification and demonstrated that the method exhibited high performance with lower computational inference costs than available alternatives. Zhang *et al.*[269] introduced LST-EMG-Net, a transformer-based hybrid architecture for EMG classification. Paper VI of this dissertation leverages the variable sequence length allowed by transformers together with a positional encoding scheme to build a single model that can be trained on EMG signals acquired using arbitrary electrode measurement geometries. To the best of my knowledge, this represents the first application of attention between channels (as opposed to temporal segments) .

Kinematic Regression

In contrast to gesture recognition, regression and regression-adjacent methods do not produce a categorical decision as output. Instead, such methods aim to map an input EMG window to some continuous numeric representation of concurrent movement intent. MCIs built on top of such methods inherently allow for proportional control, as the output values can be directly related to the force or velocity of the intended grasp or movement.

To train ML methods to perform this task, some measurable correlate of continuous motor intent must first be collected concurrently with EMG. Examples of time series used for this purpose include joint angles[270], force/torque measurements[271], or direct measurements of position and velocity of limbs[272]. These data can be acquired through means such as datagloves[273], motion capture systems[274], inertial measurement units[275], or force-sensitive resistors placed at specific joints[276]. Like with classification, a variety of regression algorithms have been evaluated for the task of estimating such values from EMG signal features (e.g. [277, 278, 279]). Feature-free

DL approaches have also been evaluated for this type of task multiple times (e.g. [257, 280, 281, 282]) with promising performance in both offline and online settings.

In prosthetics, acquiring continuous target values from amputees is significantly more challenging due to the absence of limbs. In such cases, alternative methods can be employed to obtain a suitable proxy for graded motor intent. One approach is to use contralateral limb movements to obtain ground truth targets while instructing the user to imagine performing symmetrical actions with both limbs[283]. Another simple method is provide a smoothly ramping visual prompt while instructing the user to ramp contraction intensity in conjunction with this . While such steps complicate the calibration process, they can often be worthwhile—proportional MCIs for prosthesis control has been shown to lead to higher levels of user adaptions[223].

An alternative method for achieving proportionality with ML methodology is to estimate gestures (using a classifier) and intensity of contraction separately[284, 285, 286]. Following classification, the detected gesture can be performed with velocity directly proportional to an estimate of the underlying force of contraction (typically the channel-averaged MAV value of the current EMG window, or some nonlinear transformation thereof). Paper IV [239] of this dissertation extends this framework by introducing an ANN model that, via some simple assumptions, can learn to output proportional estimates of movement intent despite being trained exclusively on categorical targets.

Motor Unit Decomposition

Motor unit decomposition refers to the process of separating HD-sEMG signals into their constituent MUAPs (see section 2.5). This separation allows for the extraction of motor units spike trains, together with MU-specific surface waveforms[287]. Practical algorithms towards this end are typically predicated on template matching or blind source separation though Independent Component Analysis with convolutive kernel compensation[288, 289]. While deep learning has been applied in attempts to extract motor unit spike trains[290], difficulties in obtaining ground truth target values has so far precluded further development.

Once obtained, spike trains provide information on the number of active motor units and their respective frequency of activation. Such factors provide considerable insight into the underlying motor intent and can as such be used to to modulate the behaviour of external devices[291]. Concretely, many methods have been introduced to derive appropriate values of force, velocity, or position from (features of) the neural drive, primarily for the application of prosthesis control.

5.4 Quantifying Performance

While the standard classification and regression metrics discussed in section 3.5 have seen significant use in MCI research, a number of special metrics have been introduced to better capture important desiderata in motion decoding. Gijsberts *et al.*[292] proposed the *movement error rate* metric as an alternative to accuracy to better handle prediction delays. Similarly, *motion selection time*, *motion completion time*, and *motion completion rate* have frequently seen use for quantitative evaluation in of prosthesis controllers[293, 294, 295]. Notably, even with application-tailored offline metrics, a number of studies have found that their correlation with performance in functional real-time tests is only weak[296, 293, 27]. As such, good offline performance should arguably be considered at best a supplement to online evaluation metrics.

Multiple frameworks exist for quantifying the real-time performance of myoelectric control algorithms. Tests based on Fitts's Law, originally conceived of as an information-theoretic relationship in the context of graphical computer user interfaces[297], have seen frequent use in studies on myoelectric control[298, 284, 299]. Similarly, in prosthetics, the Motion and Target Achievement Control tests[300, 301] are popular methods for evaluating the real-time performance of control algorithms without handling the complex minutia of physically instantiated artificial limbs.

5.5 Non-Electrical Modalities

Mechanomyography

Mechanomyography is a non-invasive technique that measures the mechanical vibrations produced by muscle contractions. These vibrations are typically detected using accelerometers, piezoelectric sensors, and/or microphones placed on the skin surface. Compared to EMG, mechanical recordings provides a more direct measure of muscle force generation and can be less susceptible to crosstalk between adjacent muscles. Furthermore, mechanomyography signals are less influenced by changes in skin-electrode contact impedance and are thus generally more stable over time[302]. Despite these advantages, MCIs based exclusively on these principles have so far seen limited adoption—mechanical surface signals have significantly lower spatial resolution compared to EMG, which limits the ability to discriminate between muscle movements associated with different motor intentions[303].

Sonomyography

Sonomyography is a technique that utilizes ultrasound imaging to visualize and analyze muscle activity. The high spatial resolution and penetration depth of ultrasound imaging allows for a detailed examination of individual muscle fibres and fascicles. By capturing real-time images of muscle contractions, sonomyography provides a non-invasive method for assessing muscle function and potentially a pathway to real-time decoding of motor intent[304]. However, ultrasound equipment is currently quite cumbersome and analysis of ultrasound images computationally expensive, making it difficult to incorporate in portable or wearable applications.

Magnetomyography

Magnetomyography is a technique that measures the magnetic fields generated by the electrical activity of muscles during contractions. This technique uses sensitive magnetometers, such as superconducting quantum interference devices (SQUIDs) or optically pumped magnetometers (OPMs), to detect the weak magnetic signals produced in conjunction with the electrical signals associated with muscle activity[305]. Due to the considerable infrastructure for cooling and shielding required to operate such devices, their usefulness for portable applications is currently highly limited.

Now this I say, brethren, that flesh and blood cannot inherit the kingdom of God.

1 Corinthians 15:50

Chapter 6

Upper Limb Prosthetics

The human hand is one of the primary interfaces by which we both sense and manipulate the world around us[306]. The dexterity granted by opposable thumbs supported by a neural basis for fine-grained motor control and feedback is not only hypothesized as a causal factor of our species unmatched technological capabilities[307], but also a necessity for many of our social and economically productive activities. In light of its importance, it is unsurprising that losing ones arm or hand through amputation is considered a traumatic event, for most on par[13] with losing approximately a decade of healthy life years[308]

An amputation can be either congenital (present at birth) or acquired, with the latter resulting from various etiological factors such as traumatic injuries, circulatory disorders, malignant tumors, and infections[309]. Amputations can be further classified according to the anatomical level at which the limb is severed. In the context of upper-limb amputations, taxonomies of amputation levels typically include (from distal to proximal): partial hand amputation, metacarpal amputations, wrist disarticulation, transradial (below elbow) amputations, elbow disarticulation, transhumeral (above elbow) amputations, and shoulder disarticulation. The differences in incidence and prelavence between amputation levels are substantial, with distal amputations being far more common than proximal amputations[310].

Following an amputation, several treatment options are available to potentially restore functionality and improve the quality of life for the affected individual. Limb replantation is a surgical procedure that involves the reconnection of a severed limb to the body[311]. While this approach may restore function and sensation, the success of the procedure is highly dependent on the type of injury and the time elapsed since amputation. Limb transplantation involves the transfer of a limb from a deceased donor to an amputee[312]. Despite offering the potential for good outcomes, this option carries risk of graft rejection and necessitates lifelong immunosuppression. Alternatively, a prosthetic limb might be used to restore a measure of functionality and appearance to the user[313].

Control of upper-limb prosthetics is currently by far the most widespread application of myoelectric interfaces in both industry and research. This chapter is a brief discussion on such devices. Section 6.1 describes how artificial limbs are typically function mechanically. Section 6.2 describes methods for controlling powered prostheses. Section 6.3 describes methods for providing sensory feedback, so far mainly in research applications. Lastly, section 6.4 describes a selection of surgical techniques that have been used to improve the functioning of prosthetics limbs.

6.1 Mechanical Actuation

Passive Prostheses

The oldest[314], most reliable, and to this day most popular[315] method of actuating a replacement limb is to not actuate it at all. *Passive prostheses* primarily serve an aesthetic purpose but can also provide non-negligible functional utility to unilateral amputees by providing support to the other hand[316]. These devices are typically designed to resemble the natural limb in shape, size, and color, although task-specific variants (hooks, tools, *etc.*) exist. While passive prostheses do not possess any active or externally powered mechanisms, they may incorporate friction-based joints, allowing manual changes in e.g. joint positions depending on the current need of the user. Compared to all available alternatives, such prostheses are low-cost, lightweight, and require minimal maintenance[317].

Body-Powered Prostheses

Body-powered prostheses are mechanically actuated devices that rely on the user's residual limb and movements of the rest of the body to provide functional control[318]. These prostheses typically consist of a harness system and a cable mechanism that translates the user's shoulder and/or torso movements into prosthetic limb motion. The two main types of body-powered prostheses are voluntary-opening and voluntary-closing systems. Voluntary-opening systems require the user to apply tension to the cable through shoulder or torso movements, which in turn opens the prosthetic terminal device (e.g., hook or hand). A spring or elastic element provides the closing force when the tension is released. Conversely, in voluntary-closing systems, the user applies tension to close the terminal device, and a spring or elastic element reopens it when tension is released. Body-powered prostheses are robust compared to active prostheses, and additionally provide users with some measure of indirect proprioceptive feedback[319].

Electrically Powered Prostheses

Electrically powered prostheses utilize small electrical motors and batteries to actuate the prosthetic device and are usually controlled by EMG signals from residual muscles. The conventional type of powered hand prosthesis is the single-DoF gripper. This type of prosthesis is easy to control and provides a basic open-and-close mechanism for the prosthetic fingers, allowing the user to perform simple grasping and holding tasks[320].

Powered *multifunctional* prostheses, albeit typically both expensive, heavy, and fragile (compared to aforementioned alternatives), are capable of simultaneously and independently actuating multiple degrees of freedom. These devices typically feature 10 motors, which can be compared to the 27 inherent DoFs present in the human hand. This increased level of complexity allows for the execution of more intricate and coordinated movements, mimicking the range of motion exhibited by the natural hand to a greater extent. However, more dextrous hands typically exhibits weaker grasp forces, as space constraints induced by multiple motors necessarily reduces their allowed size and by extension force[321]. In so-called *underactuated* designs, each actuator (motor) is used to drive more than one DoFs through tendon or linkage-based mechanisms, limiting weight and cost while retaining the ability to perform a larger range of motions than would be possible with a one-to-one relationship between motors and DoFs[322].

6.2 Artificial Efference

In body-powered prostheses, the actuation mechanism is directly coupled to the control of the device. For electrically powered prosthetics however, the control signal must be supplied exogenously. The by far most common choice of control signal is EMG collected from remnant muscles of the residual limb.

Direct Control

Direct control makes makes use of pairs of sEMG electrodes placed on antagonistic muscle pairs located superficially in the residual limb of the user. The difference in some measure of amplitude (typically rectified and low-pass filtered EMG) between the signals from a single pair are then mapped directly to the force driving a single DoF. Due to the limited number of suitable muscle pairs in even a fully intact arm, direct control can only accomodate a handful of simultaneously controllable DoFs. For multifunctional prostheses, additional DoFs must be controlled sequentially by use of auxiliary protocols, e.g. based on co-contraction or non-EMG inputs for DoF switching.

Direct control is by far the most common control method in commercially available prostheses—it is both highly robust very simple to implement on embedded hardware. However, this approach has obvious limitations. Direct control is not natural or intuitive, as the muscle activation patterns required to perform some motion do not correspond to the natural pattern that would correspond to the same motion with an intact limb. This lack of intuitive leads to increased cognitive burden, and has been proposed[323] as one of many explanation for the high abandonment rate of myoelectric prostheses[315, 313].

Pattern Recognition Control

Pattern recognition control functions by the principles outlined in section 5.3. In brief, sampled EMG is processed in real-time to extract features from a continuously updated signal window; features are in turn is fed to a decoding algorithm running on embedded hardware. The algorithm returns an estimate of the concurrent motion intent, which is thereafter converted to a motion command that is mechanically actuated by the prosthesis. The main advantage of this approach over the simpler direct control paradigm is the possibility of natural control (detected patterns can be mapped to the same movement that the pattern corresponds to in a healthy individual) and multiarticulate control (an arbitrary number of DoFs can be involved in when performing a detected motion). Simultaneity and proportionality of control (as defined in section 5.1) are compatible with, but do not follow automatically from, this approach.

As is apparent throughout chapter 5, the research literature abounds with proposals for advanced signal processing and ML methods for intuitive EMG decoding. However, very few have so far been translated into clinically viable systems, and the proportion of users that control their myoelectric prosthesis via pattern recognition is still low[324]. Examples of commercially available systems that incorporate pattern recognition include Myo Plus by Ottobock[325], Sense by Infinite Biomedical Technologies[326], and Coapt[327]. While none of the aforementioned companies have disclosed the exact algorithm(s) comprising the processing pipeline, it is reasonable to assume (based partially on the calibration protocols of these systems, which only instructs the user to repeat a set of prespecified grasps and/or gestures) that nothing more advanced than classificatory gesture recognition (e.g. LDA on some compute-efficient feature set) is involved.

6.3 Artificial Afference

Researchers and users alike frequently stress that sensory feedback is simultaneously (i) one of the most important and (ii) exceedingly rate in commercially available prostheses[328]. Together with a functioning controller, the integration of sensory feedback constitutes a *closed-loop* system[329], allowing for motor behaviour more similar to that characteristic of a healthy individual. This section covers some methods in contemporary research that have been proposed to provide actionable feedback to the user.

Sensor Modalities

Sensor modalities are the methods by which prosthetic devices acquire information from the environment or the user's intent. Various sensor types have been explored in upper limb prostheses, examples of which include:

Force and pressure sensors measure the magnitude of force applied to an object during manipulation. These sensors can be integrated into prosthetic fingers or grasp surfaces, allowing the user to receive feedback on grip strength and object interaction[330].

Thermal sensors measure the temperature at various points on the surface of the prosthesis . Like the thermoreceptors of regular skin, they can be useful to the user in the management of objects that are very hot old cold[331].

Angle and position sensors measure what it says on the tin. This data can provide useful proprioceptive information on the current state and posture of the prosthesis[332].

Feedback Mechanisms

Feedback mechanisms relay sensor information to the user. Several feedback methods, both invasive and non-invasive, have been proposed and explored for upper limb prostheses. Some examples include:

Vibrotactile feedback involves the use of small vibrating motors (tactors) placed on the user's residual limb or other body locations. The tactors generate vibrations in response to sensor data, allowing the user to interpret force, pressure, or other sensory information[333].

Electrical stimulation delivers electrical current through electrodes placed on the user's skin to generate a perceptual sensation. The intensity, frequency, and spatial distribution of the stimulation can be modulated to convey different types of sensory information[334].

Auditory feedback uses sound to convey information about the prosthetic limb's status or environmental interaction. For example, different tones or synthesized sounds can represent force levels, grip types, or other relevant information[335].

In current commercial systems, the normal functioning of a powered prosthesis has been noted to provide some measure of useful *incidental feedback*. For instance, the sound emitted by motors during operation allows users to gauge the speed of their prosthetic limb. Similarly, subtle vibrations can indicate the force exerted by a prosthetic hand while gripping an object[336].

6.4 Surgical Techniques

Osseointegration

Osseointegration[337] refers to the direct connection of a prosthesis to the patient's residual bone, creating a stable and long-lasting interface between the device and the skeletal system. This technique involves the surgical implantation of a titanium fixture into the medullary canal of the residual bone. The fixture fuses with the bone through a process called biological fixation, providing an attachment for the prosthetic limb[338].

Osseointegration offers several advantages over traditional socket prostheses, including a more natural range of motion and reduced skin irritation. Additionally, osseointegrated prostheses enable direct transmission of mechanical loads to the skeleton, allowing for better proprioceptive feedback. However, as always, surgical interventions are associated with risks. Osseaointegrated implants can be rejected or mediate infections and result in fractures if too loaded[339].

Targeted Muscle Reinnervation

Targeted muscle reinnervation (TMR) is a surgical procedure designed to improve the control of myoelectric prostheses by increasing the number and spatial separation of available EMG channels[340, 300]. TMR involves transferring the residual nerves from the amputated limb to other muscles—which have lost their their biomechanical

function in the absence of a limb—that are then used to generate EMG signals for prosthesis control.

During TMR, the severed nerves are coapted to motor nerves in the target muscles (typically in the chest or upper arm), enabling the reinnervated muscles to produce distinct EMG signals in response to the user's intended movements. These signals can then be detected by surface or implanted EMG electrodes and used to control the prosthetic limb[341].

Implanted Electrodes

Implanted electrodes[342] are used to directly record electrical activity from muscles, providing a more stable and spatially precise interface for prosthetic control compared to surface electrodes. These electrodes are surgically implanted into the muscle tissue, allowing for a more robust connection over time, negligible cross-talk, and improved signal-to-noise ratio.

There are different types of implanted EMG electrodes, including intramuscular, epimysial, and nerve cuff electrodes. Intramuscular electrodes are inserted into the muscle fibres, while epimysial electrodes are placed on the surface of the muscle. Nerve cuff electrodes, on the other hand, are wrapped around the peripheral nerves, enabling the detection of signals from multiple muscles simultaneously.

Part II

Included Papers

Chapter 7

Summary of Included Papers

This chapter contains brief summaries of the papers contained in this dissertations: 5 published journal papers (papers I to V), 1 as of yet unpublished manuscript (paper VI), and 2 published conference papers (papers VII and VIII). To varying extents, all papers deal with applying ML and DL methods for the purpose of qualitatively or quantitatively improving the decoding performance of myoelectric interfaces. According to the author, the main take-home messages of the studies can be stated as follows: Both papers I and VI demonstrate the capabilities enabled by overparametrized models and large amounts of data and the possibility of leveraging such capability in the context of muscle-computer interfacing. Conversely, paper II suggests that DL models can function adequately even in contexts where excessive computational resources are unavailable if appropriate design considerations are complied with. Paper III showcases the potential in a tailored approach toward ANN training in order to achieve desirable properties of the decoding interface. Paper IV, although not demonstrating or suggesting anything in particular, provides a large dataset that could prove useful for other ML-based methods (as it indeed did for the study in paper VI). Paper VII concisely illustrate the strengths (and some weaknesses) of recurrent models for motion decoding. Lastly, paper VIII provides the earliest of indications that the 'Big Data' regime can be extended to user-compatible myoelectric interfaces by substituting a large fraction of the required training data with simulacra.

Paper 1:
Extraction of multi-labelled movement information from the raw HD-sEMG image with time-domain depth

This paper details the development and implementation of an MCI based on high-density surface electromyogram (HD-sEMG) signals. Traditional MCI systems suffer from limitations in terms of control robustness and versatility, mainly due to the complexity of neuromuscular processes and the difficulty in creating a mapping between sEMG signals and movement commands. We proposed an alternative approach by modeling hand gestures as combinations of simpler ”basis” movements, utilizing multi-label machine learning. This approach has the potential to offer scalability, stability, and an increased range of performable movements compared to conventional multi-class (single-label) classification techniques.

We selected 16 labels representing flexion and extension of all digits and the wrist, thumb abduction and adduction, and wrist pronation and supination. These labels were chosen based on the assumptions that they engage different forearm muscles or muscle compartments and, when combined, can capture the major DoFs possessed by the human hand. A deep learning classifier in the form of a CNN (see Figure 7.1) was implemented to detect these movement labels in the HD-sEMG signal.

The study introduces a novel method of utilizing structured 3-dimensional input volumes composed of stacked time slices, each representing the instantaneous muscle state, to decode movement intent.

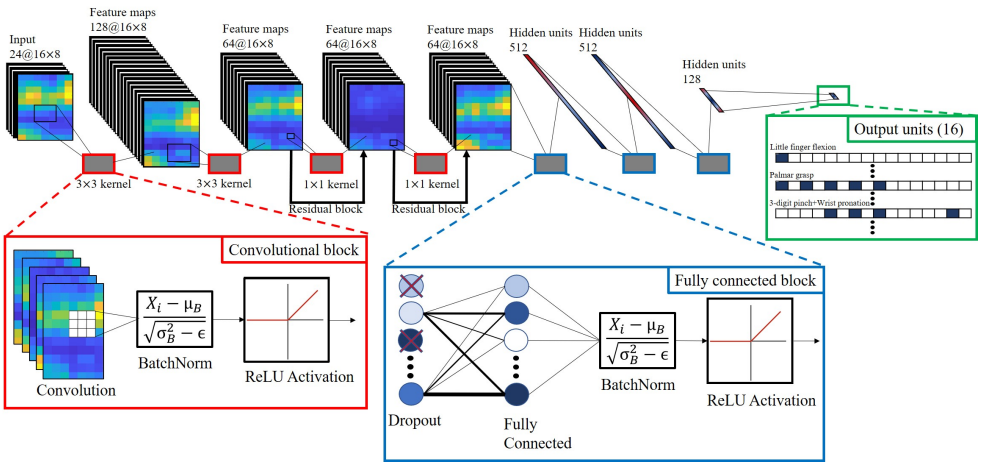


Figure 7.1: The CNN architecture developed and evaluated in paper I.

Paper II:
Automatic discovery of resource-restricted Convolutional Neural Network topologies for myoelectric pattern recognition

This paper proposes a method for automatically discovering CNN topologies that are optimized specifically for myoelectric pattern recognition tasks while being restricted by limited computational resources. We use a genetic algorithm to evolve CNN topologies that are optimized for both accuracy and computational efficiency. The proposed method is evaluated on two publicly available datasets, and the results show that the evolved CNNs outperform existing state-of-the-art methods in terms of accuracy while being computationally efficient.

The proposed method is evaluated on two publicly available datasets, and the results show that it can discover CNN architectures that achieve state-of-the-art performance while satisfying the resource constraints.

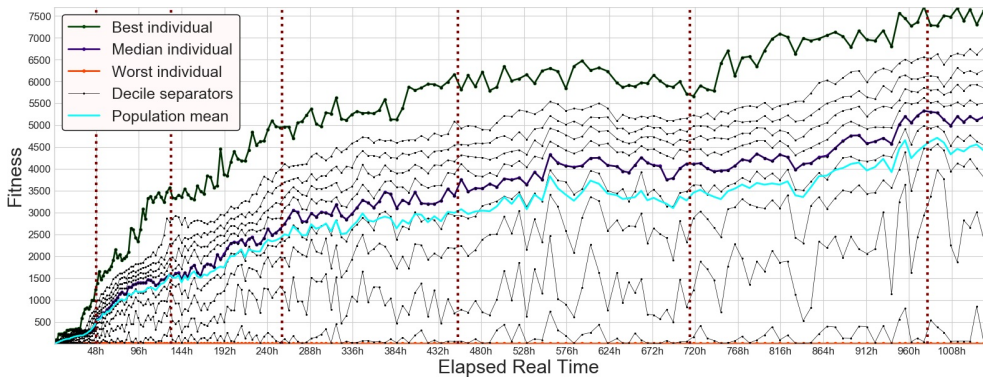


Figure 7.2: The fitness distribution over the population of myoelectric decoding CNNs as as a function of runtime during the evolutionary search.

Paper III:
Learning regularized representations of categorically labelled surface EMG enables simultaneous and proportional myoelectric control

This paper evaluates a new framework, termed Myoelectric Representation Learning (MRL) for the natural control of upper extremity prostheses using sEMG signals. The framework aims to provide simultaneous and proportional control over multiple DoFs in real-time. MRL uses a multitask neural networks (see Figure 7.3) and domain-informed regularization to find nonlinear mappings from forearm sEMG to multivariate and continuous encodings of hand- and wrist kinematics.

The study collected sEMG from 20 healthy subjects and used the data to calibrate two control interfaces, each with two output DoFs. One was built using the MRL framework, and the other used a standard pattern recognition framework based on LDA for sake purpose of comparison. The online performances of both interfaces were assessed using a Fitts's law type test, generating five performance metrics. Stability was evaluated by conducting identical tests without recalibration seven days after the initial experiment.

The results showed a significant advantage for MRL over LDA in all performance metrics, indicating that MRL can provide superior real-time performance compared to the current status quo pattern recognition. There were no significant effects on any metric detected for neither session nor interaction between method and session, suggesting that neither method deteriorated significantly in control efficacy.

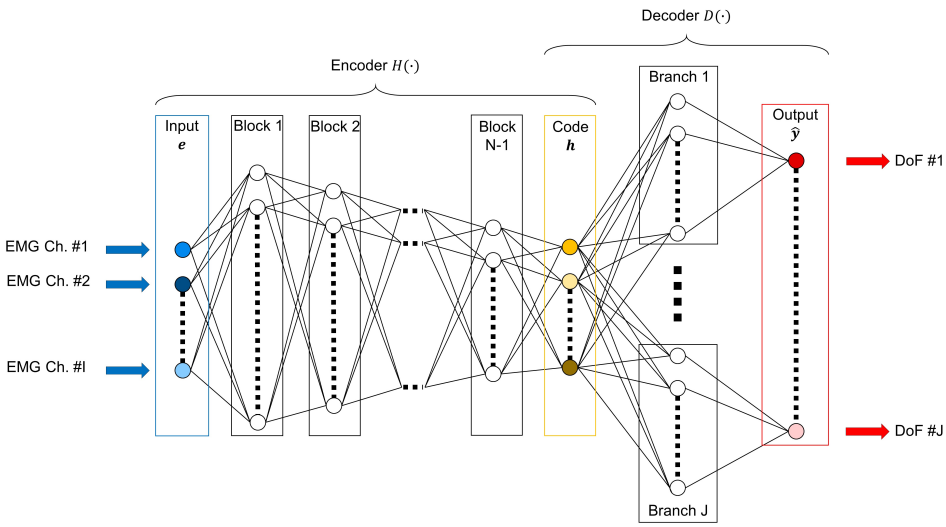


Figure 7.3: The proposed multitask ANN that allows for simultaneous and proportional control from training with categorical target labels.

Paper IV:
A database of high-density surface electromyogram signals comprising 65 isometric hand gestures

This study presents a database of HD-sEMG signals to support the development of robust and versatile electromyographic control interfaces for prosthetic hands. The signals were recorded from the forearms of 20 able-bodied volunteers who performed 65 different hand gestures in an isometric manner (experiment setup shown in Figure 7.4). The data aims to contribute to better myoelectric decoding schemes and enable more dexterous control interfaces for prosthetic hands.

The database contains 128 channels of sEMG data and synchronous movement and joint force information. The 65 hand movements were interpreted as compounds of 16 basis movements, capturing the major DoFs of the hand and wrist. This decomposition may allow for multi-label machine learning approaches to develop more dexterous control interfaces. The inclusion of wrist and digit forces enables the exploration of proportional control for prosthetic hands.

Technical validation of the data included frequency spectra analysis, channel cross-correlations, and detection of poor skin-electrode contacts. Results indicate that the acquired signals were of high quality, with limited common noise and crosstalk. The database is intended to facilitate the development of novel methods and allow for fair comparison between different approaches.

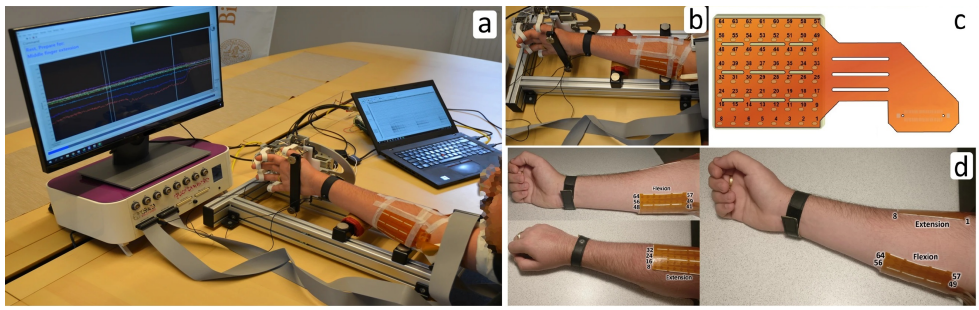


Figure 7.4: The recording setup used for the collection of the database. (a) and (b): The participant's hand positioned inside the force measurement device. (c): High-density electrode (skin top view). (d): Position of flexion and extension electrodes on the arm of a subject.

Paper v:
End-to-end estimation of hand-and wrist forces from raw intramuscular EMG signals using LSTM networks

This study investigates the use of deep learning models based on the LSTM architecture for processing iEMG signals, aiming to improve motor intent decoding for controlling motorized prostheses in transradial amputees. Three LSTM models were evaluated, including One-to-One, All-to-One, and All-to-All strategies between input iEMG channels and output forces. The models were assessed using a dataset of six iEMG channels with concurrent force measurements from 14 subjects. Results showed that all LSTM strategies significantly outperformed the baseline feature-based linear control regression method, suggesting that recurrent neural networks can efficiently transform raw forearm iEMG signals into representations correlating with forces exerted at the hand. Moreover, the All-to-All and All-to-One strategies demonstrated better performance than the One-to-One strategy, indicating that iEMG from muscles not directly actuating the relevant degree of freedom can provide contextual information to aid in decoding motor intent (see Figure 7.5).

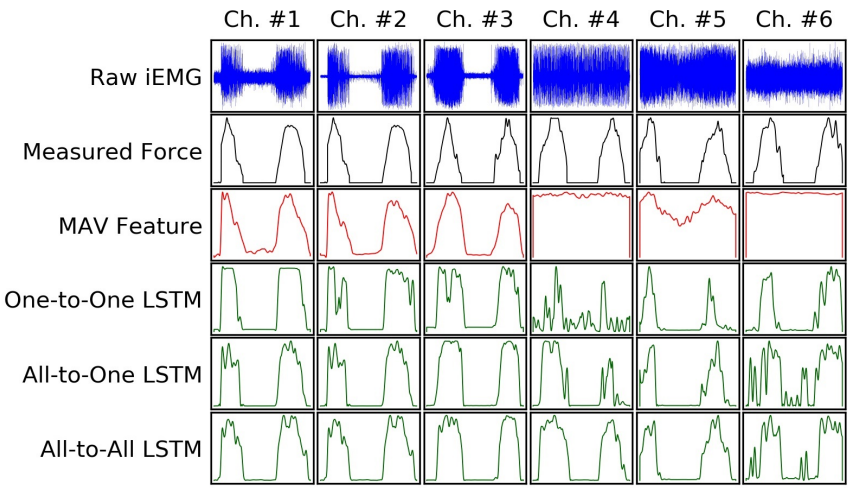


Figure 7.5: Example of iEMG presented to, and output force estimates produced by, all regression models investigated in paper v.

Paper VI: Calibration-free myoelectric decoding

In this study, we developed a ML-based framework for efferent myoelectric interfacing that does not require the collection of new user-specific training data. We designed a geometry-aware Transformer-based model architecture that can handle input sEMG windows from arbitrary electrode configurations and outputs a generalized intent representation. We initially pretrained our model on 30 publicly available sEMG databases, comprising 510 subjects and 108 unique gestures. Our user-agnostic models were then evaluated and finetuned on the NinaPro DB2 and DB3 databases. The results indicated that our calibration-free intent decoding approach performs competitively with user-specific LDA models, but requires further modifications to fit on embedded hardware and may need complementary approaches for prosthetic applications.

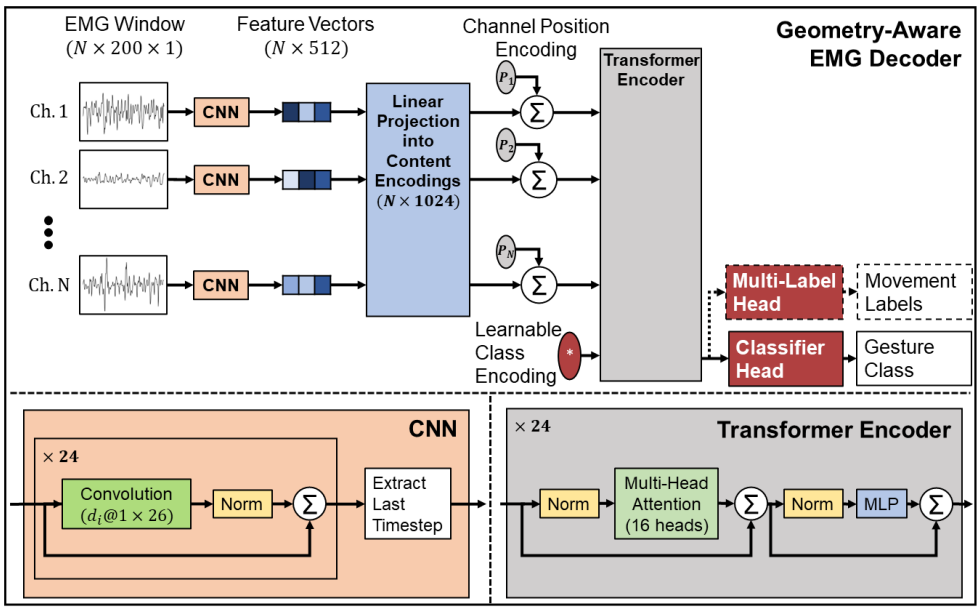


Figure 7.6: The geometry-agnostic, encoder-only transformer model introduced in paper VII. By extracting learnable feature vectors from all channels and subsequently feeding the sequence of channel-wise (position-encoded) feature vectors to a transformer, the model can operate on EMG signal windows from arbitrary measurement geometries.

Paper VII:
Exploiting the intertemporal structure of the upper-limb sEMG: comparisons between an LSTM network and cross-sectional myoelectric pattern recognition methods

This paper presents a comparative study between traditional cross-sectional myoelectric pattern recognition methods and a classifier built on the natural assumption of temporal ordering by utilizing an LSTM neural network.

The study found that the LSTM approach outperforms traditional gesture recognition techniques which are based on cross-sectional inference. These findings held both when the LSTM classifier operated on conventional features and on raw sEMG and for both healthy subjects and transradial amputees.

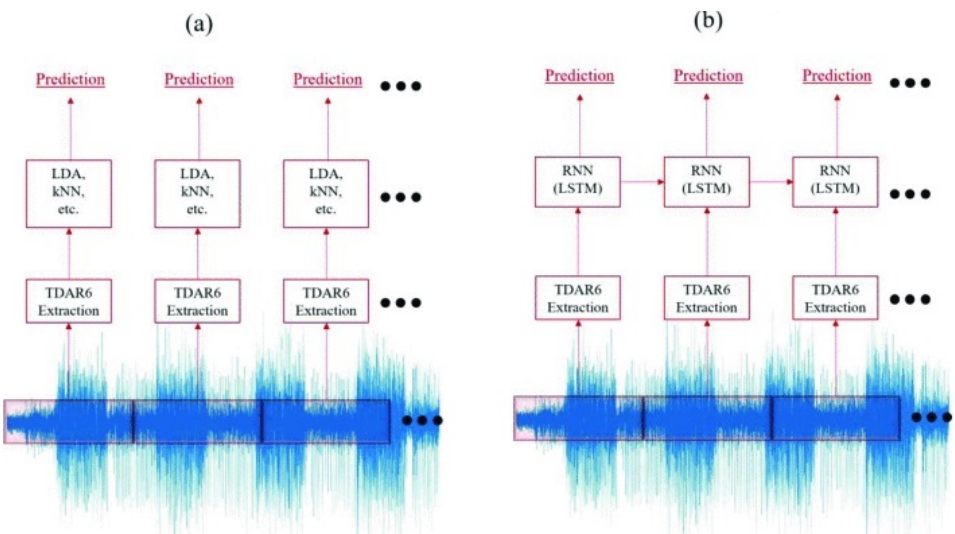


Figure 7.7: The myoelectric pattern recognition procedure for (a) the cross-sectional classifiers and (b) the LSTM network time series classifier.

Paper VIII:
Can deep synthesis of EMG overcome the geometric growth of training data required to recognize multiarticulate motions?

This paper proposes a technique intended to circumvent the combinatorial explosion of training data required for pattern recognition approaches to myoelectric control. The technique involves using EMG windows from 1-DoF motions as input and EMG windows from 2-DoF motions as targets to train generative DL models to synthesize EMG windows for to multi-DoF motions. Once trained, such models can be used to complete datasets consisting of only 1-DoF motions, enabling calibration protocols that scale linearly with the number of DoFs.

The paper evaluated synthetic EMG produced in this way via a classification task using a database of forearm surface EMG collected during 1-DoF and 2-DoF motions. Multi-output classifiers were trained on either (I) real data from 1-DoF and 2-DoF motions, (II) real data from only 1-DoF motions, or (III) real data from 1-DoF motions appended with synthetic EMG from 2-DoF motions. When tested on data containing all possible motions, classifiers trained on synthetic-appended data (III) significantly outperformed classifiers trained on 1-DoF real data (II), although significantly underperformed classifiers trained on both 1- and 2-DoF real data.

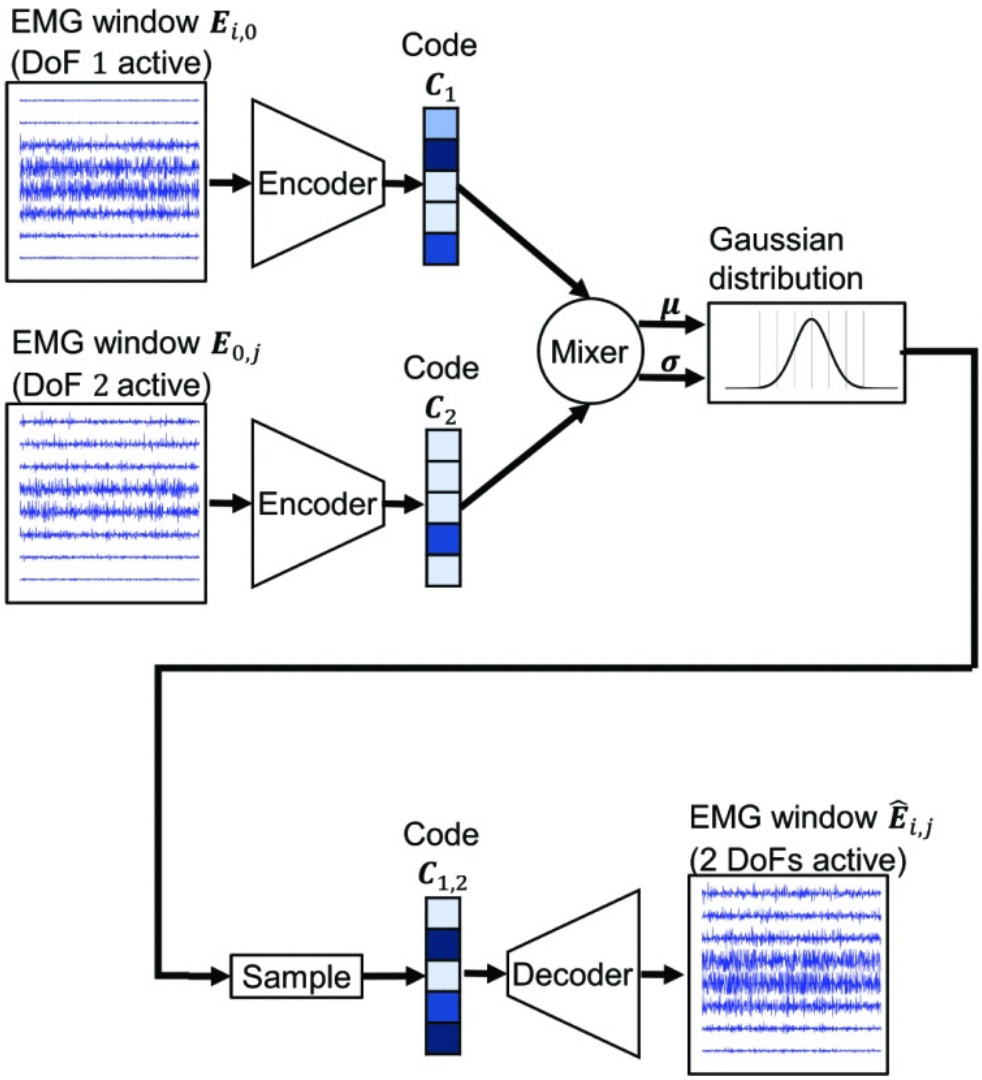


Figure 7.8: Schematic overview of the synthesizer model introduced in paper VIII.

Chapter 8

Closing Remarks

In 1773, John Hunter made the first systematic attempt at explaining the ability of the common torpedo fish to produce extracorporeal electrical fields, inadvertently laying the foundation for our modern understanding of electrochemistry[343]. Exactly 250 years later, consensus holds that *electric organs* have evolved independently multiple times in the animal kingdom, with at least five instances originating from skeletal muscles[344]. Using the right tools, humans may soon be able to join our distant fish relatives in turning muscles from organs of movement to organs for *communication*. Electrical signals produced by muscles are not only strong, making them simple to detect and record, but correlates to a substantial degree with volitionally produced motor intent. As a consequence, the research literature abounds with proposals (some of which comprise this dissertation) for transducing myoelectricity into control signals for external devices.

By circumventing the dissipative properties of tissue, iEMG provides high signal quality and spatial precision, making latent intent decoding from signals relatively simple—mapping signal envelopes directly to output DoFs is often sufficient. As was discovered in paper v of this dissertation, more sophisticated processing may be used to improve iEMG decoding performance even further. Even so, for many practical applications, the invasive and delicate measurement setups required for iEMG can be obstructive, particularly if intended to function in mobile setting. In such contexts, sEMG holds a clear edge as a risk-free alternative, in particular for the long-term recordings required for muscle-computer interfacing. Without access to spatially precise recordings, HD-sEMG (involved in papers i and iv) can be construed as a way of providing sufficient information via overwhelming quantity. As such signals are inherently convoluted, involving the superposition of MUAPs from multiple muscles, methods for extracting and aggregating information from across measurement locations are necessary to infer anything beyond the simplest of output commands.

While careful feature engineering has yielded accurate classificatory motor decoding models, Deep Learning has recently emerged as a promising approach for end-to-end

extraction of intent from raw sEMG signals. As has been corroborated here (papers I, II, and VI) and elsewhere, deep models can learn signal representations directly from raw data; as in many other domains, the ability of ANNs to learn something akin to intuitions in domains where humans have none has resulted in decoding performances better than those of previous, handcrafted alternatives. While results so far have been highly promising, improvements in this area seems eminently possible both on the algorithmic side and on the application-specific side.

Accurate decoding of motions is necessary for MCIs to be functional, but is likely not sufficient. To be compatible with the wide array of applications proposed in human-computer interfacing, decoding algorithms should ideally exhibit some degree of proportionality and simultaneity. The findings of paper I indicates that simultaneous control over a large number of DoFs is possible in principle. Future work in multi-label decoding should focus on investigating the potential for generalization to novel label combinations. Paper III demonstrated the feasibility of simultaneous and proportional control without a force-varied training dataset by using domain-informed regularization. A number of extensions of this approach seem plausible. The algorithm of paper III worked in the context of sEMG signal envelopes, drastically reducing the information available to the network. A valuable contribution to this specific research agenda would be to device more sophisticated regularization objectives that still gives the decoding algorithm access to the full spectral content of input signals.

It is debatable to what extend the specific choice of ANN model architecture plays a role in determining final performance[201]. More certain is the fact that model architecture matters a lot for the *size* of any given implementation, and by extension its (for MCIs crucially important) ability to fit in embedded systems. In paper II, an evolutionary algorithms was applied to search for optimal CNN topologies and constraining the search space to exclude prohibitively compute-intensive architectures. We showed that lightweight application-specific models can be developed that maintain high performance in comparison to previous, general architectures. In light of this finding, more meta-optimization seems warranted—while such considerations are strangely rare at the forefront of contemporary research in artificial intelligence, it is arguably more important in the data- and compute-limited context of sEMG signals.

Achieving inter-user models is a significant challenge in the development of muscle-computer interfaces, but of unparalleled importance, especially for commercial and industrial applications. In paper VI, we introduced a geometry-aware Transformer-based model architecture that can operate on input sEMG windows collected from arbitrary electrode configurations and output generalized intent representations, enabling the creation of user-agnostic models with performance

comparable to user-specific models in certain scenarios. However, the accuracies of these models are still subpar, requiring further refining work and, perhaps more importantly, data. Methods for collecting large unlabelled EMG datasets could be a promising future research direction to truly leverage the seeming lack of upper limits on performance in large ANN models[345]. Keeping in mind the scarcity of memory and processing power characteristic of embedded systems, such endeavour must be matched by attempts at knowledge distillation.

Expanding the inter-user framework to amputees presents significant challenges due to the smaller population size and variability in residual limb characteristics. These factors can significantly impact the performance of MCIs and limit the applicability of generalized models. Consequently, there is a need for tailored approaches that accommodate the specific needs of amputees in the development of prosthetic applications. One complementary approach was investigated in paper VIII, where a generative deep learning framework was proposed to synthesize EMG data for multi-degree-of-freedom motions, potentially enabling simpler calibration protocols. Additional exploration into the use of transfer learning and/or domain adaptation techniques that leverage pre-trained models on non-amputee data and fine-tuning them using a smaller dataset from amputees seems likely to yield fruit.

Notes

1 Assuming the non-existence of purported phenomena such as psychokinesis (see e.g. [346] for a critical review).

2 For example, the contraction of a residual biceps muscle could be used to control the flexion of a prosthetic elbow, while the contraction of a residual triceps muscle could be used to control its extension.

3 The opaqueness of biological systems has in fact been theorized[347] to be evolutionarily adaptive: In an environment full of pathogens set on manipulating host behaviour, inscrutable complexity can in fact be a viable defence mechanism. This hypothesis, albeit intriguing, lies far outside the scope of this dissertation.

4 Neuroscience might reasonably be construed as also belonging in this category.

5 The prompt used was "Write a pithy introduction for a chapter on Machine Learning".

6 If a stochastic target function is considered, the expression for the total error should also include an irreducible error term.

7 It is worth noting that many unsupervised problems can be formulated as supervised learning problems via self-supervised learning, as is discussed in section 3.2.

8 This also explains the need for a nonlinear activation function following the convolution operator in eq. 4.21—without it, a sequence of convolutional layers would only ever be able to learn linear mappings.

9 The terminology stems from the field of natural language processing, where e.g. autoregressive self-supervised training objectives are often used, necessitating *causal masking* of the input sequence of the decoder (cf. [348]). The term encoder-only used to describe (for example) vision transformers signifies that no causal masking is performed

10 Some recent work dispute the claim that normalization facilitates training by reducing internal covariate shift and instead propose alternative explanations for its tangible effect on convergence[349, 350, 351].

11 The ostensible power of depth is sometimes informally motivated through *compositionality*[352]. In image classification, many high-level concepts in can naturally be defined through their compositional relationships with lower-level concepts—for example, faces might be construed as consisting of eyes, noses, and mouths, whereas these in turn can be understood as compositions of more basic shapes.

12 Exemplified in works like *Neuromancer*, *Blindsight* and *The Matrix*, among many others.

13 In terms of lost quality-adjusted life years.

References

[1] J. Riggio, J. E. M. Baillie, S. Brumby, E. Ellis, C. M. Kennedy, J. R. Oakleaf, A. Tait, T. Tepe, D. M. Theobald, O. Venter, J. E. M. Watson, and A. P. Jacobson, "Global human influence maps reveal clear opportunities in conserving earth's remaining intact terrestrial ecosystems," *Global Change Biology*, vol. 26, no. 8, pp. 4344–4356, 2020.

[2] L. Resnik, M. Borgia, B. Silver, and J. Cancio, "Systematic review of measures of impairment and activity limitation for persons with upper limb trauma and amputation," *Archives of Physical Medicine and Rehabilitation*, vol. 98, no. 9, pp. 1863–1892.e14, 2017.

[3] S. Waldert, "Invasive vs. non-invasive neuronal signals for brain-machine interfaces: will one prevail?" *Frontiers in Neuroscience*, vol. 10, 2016.

[4] G. Pfurtscheller and C. Neuper, "Motor imagery and direct brain-computer communication," *Proceedings of the IEEE*, vol. 89, no. 7, pp. 1123–1134, 2001.

[5] A. Singh, A. A. Hussain, S. Lal, and H. W. Guesgen, "A comprehensive review on critical issues and possible solutions of motor imagery based electroencephalography brain-computer interface," *Sensors*, vol. 21, no. 6, 2021.

[6] N. Naseer and K.-S. Hong, "fNIRS-based brain-computer interfaces: a review," *Frontiers in Human Neuroscience*, vol. 9, Jan. 2015. [Online]. Available: https://doi.org/10.3389/fnhum.2015.00003

[7] E. A. Felton, R. G. Radwin, J. A. Wilson, and J. C. Williams, "Evaluation of a modified fitts law brain–computer interface target acquisition task in able and motor disabled individuals," *Journal of Neural Engineering*, vol. 6, no. 5, p. 056002, 2009.

[8] N. Nappenfeld and G.-J. Giefing, "Applying fitts' law to a brain-computer interface controlling a 2d pointing device," in *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2018, pp. 90–95.

[9] D. Dindo, N. Demartines, and P.-A. Clavien, "Classification of surgical complications," *Annals of Surgery*, vol. 240, no. 2, pp. 205–213, Aug. 2004.

[10] A. B. Schwartz, "Cortical neural prosthetics," *Annual Review of Neuroscience*, vol. 27, no. 1, pp. 487–507, 2004, pMID: 15217341.

[11] G. Hong and C. M. Lieber, "Novel electrode technologies for neural recordings," *Nature Reviews Neuroscience*, vol. 20, pp. 330–345, 2019.

[12] L. R. Hochberg, D. Bacher, B. Jarosiewicz, N. Y. Masse, J. D. Simeral, J. Vogel, S. Haddadin, J. Liu, S. S. Cash, P. van der Smagt, and J. P. Donoghue, "Reach and grasp by people with tetraplegia using a neurally controlled robotic arm," *Nature*, vol. 485, no. 7398, pp. 372–375, May 2012.

[13] E. C. Leuthardt, D. W. Moran, and T. R. Mullen, "Defining surgical terminology and risk for brain computer interface technologies," *Frontiers in Neuroscience*, vol. 15, Mar. 2021.

[14] F. Harris, G. Lee, T. C. Ting, S. Rubin, B. Gaston, and G. Hu, "Impact of computing on the world economy: A position paper," in *21st ISCA International Conference on Computer Applications on Industry and Engineering (CAINE 2008)*, 2008.

[15] F. Donders, "On the speed of mental processes," *Acta Psychologica*, vol. 30, pp. 412–431, 1969.

[16] M. Cerf and M. Garcia-Garcia, Eds., *Consumer Neuroscience*, ser. The MIT Press. London, England: MIT Press, Nov. 2017.

[17] X. Navarro, T. B. Krueger, N. Lago, S. Micera, T. Stieglitz, and P. Dario, "A critical review of interfaces with the peripheral nervous system for the control of neuroprostheses and hybrid bionic systems," *Journal of the Peripheral Nervous System*, vol. 10, no. 3, pp. 229–258, 2005.

[18] R. Merletti and P. A. Parker, *Introduction to Surface Electromyography*. Jones & Bartlett Learning, 2011.

[19] D. Farina and A. Holobar, "Human-machine interfacing by decoding the surface electromyogram," *IEEE Signal Processing Magazine*, vol. 32, no. 1, pp. 115–120, 2015.

[20] D.-H. Kim, N. Lu, R. Ma, Y.-S. Kim, R.-H. Kim, S. Wang, J. Wu, S. M. Won, H. Tao, A. Islam, K. J. Yu, T. il Kim, R. Chowdhury, M. Ying, L. Xu, M. Li,

H.-J. Chung, H. Keum, M. McCormick, P. Liu, Y.-W. Zhang, F. G. Omenetto, Y. Huang, T. Coleman, and J. A. Rogers, "Epidermal electronics," *Science*, vol. 333, no. 6044, pp. 838–843, 2011.

[21] V. Gohel and N. Mehendale, "Review on electromyography signal acquisition and processing," *Biophysical Reviews*, vol. 12, pp. 1361–1367, 2020.

[22] K. J. Zuo and J. L. Olson, "The evolution of functional hand replacement: from iron prostheses to hand transplantation," *Plastic surgery (Oakv)*, vol. 22, no. 1, pp. 44–51, 2014.

[23] J. Paciga, P. Richard, and R. Scott, "Error rate in five-state myoelectric control systems," *Medical and Biological Engineering and Computing*, vol. 18, pp. 287–290, 1980.

[24] A. Saikia, S. Mazumdar, N. Sahai, S. Paul, D. Bhatia, S. Verma, and P. K. Rohilla, "Recent advancements in prosthetic hand technology," *Journal of medical engineering & technology*, vol. 40, no. 5, pp. 255–264, 2016.

[25] B. Hudgins, P. Parker, and R. Scott, "A new strategy for multifunction myoelectric control," *IEEE Transactions on Biomedical Engineering*, vol. 40, no. 1, pp. 82–94, 1993. [Online]. Available: https://doi.org/10.1109/10.204774

[26] C. Castellini and P. van der Smagt, "Surface EMG in advanced hand prosthetics," *Biological Cybernetics*, vol. 100, no. 1, pp. 35–47, Nov. 2008.

[27] E. Scheme and K. Englehart, "Electromyogram pattern recognition for control of powered upper-limb prostheses: State of the art and challenges for clinical use," *The Journal of Rehabilitation Research and Development*, vol. 48, no. 6, p. 643, 2011. [Online]. Available: https://doi.org/10.1682/jrrd.2010.09.0177

[28] N. Jiang, S. Dosen, K.-R. Muller, and D. Farina, "Myoelectric control of artificial limbs—is there a need to change focus?[in the spotlight]," *IEEE Signal Processing Magazine*, vol. 29, no. 5, pp. 152–150, 2012.

[29] I. Kyranou, S. Vijayakumar, and M. S. Erden, "Causes of performance degradation in non-invasive electromyographic pattern recognition in upper limb prostheses," *Frontiers in Neurorobotics*, vol. 12, 2018.

[30] A. Phinyomark and E. Scheme, "Emg pattern recognition in the era of big data and deep learning," *Big Data and Cognitive Computing*, vol. 2, no. 3, p. 21, 2018.

[31] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, pp. 1798–1828, 08 2013.

[32] M. B. Ganapini, M. Campbell, F. Fabiano, L. Horesh, J. Lenchner, A. Loreggia, N. Mattei, F. Rossi, B. Srivastava, and K. B. Venable, "Thinking fast and slow in ai: the role of metacognition," in *International Conference on Machine Learning, Optimization, and Data Science*, 2021.

[33] E. R. Kandel, J. H. Schwartz, T. M. Jessell, S. A. Siegelbaum, and A. J. Hudspeth, *Principles of neural science, fifth edition*, 5th ed. New York, NY: McGraw-Hill Medical, Dec. 2000.

[34] H. K. Inagaki, S. Chen, K. Daie, A. Finkelstein, L. Fontolan, S. Romani, and K. Svoboda, "Neural algorithms and circuits for motor planning," *Annual Review of Neuroscience*, vol. 8, pp. 249–271, 07 2022.

[35] A. M. Valevicius, Q. Boser, E. Lavoie, G. Murgatroyd, P. Pilarski, C. Chapman, A. H. Vette, and J. Hebert, "Characterization of normative hand movements during two functional upper limb tasks," *PLoS One*, vol. 13, 06 2018.

[36] E. N. Marieb and K. Hoehn, *Human Anatomy & Physiology, Global Edition*, 11th ed. London, England: Pearson Education, Dec. 2018.

[37] R. Lewis, K. E. Asplin, G. Bruce, C. Dart, A. Mobasheri, and R. Barrett-Jolley, "The role of the membrane potential in chondrocyte volume regulation," *Journal of Cellular Physiology*, vol. 226, no. 11, pp. 2979–2986, Aug. 2011.

[38] D. Bucher and P. A. V. Anderson, "Evolution of the first nervous systems – what can we surmise?" *Journal of Experimental Biology*, vol. 218, no. 4, pp. 501–503, Feb. 2015.

[39] J. Gordon Betts, P. Desaix, E. W. Johnson, J. E. Johnson, O. Korol, D. Kruse, B. Poe, J. Wise, M. D. Womble, and K. A. Young, *Anatomy and physiology*, 2022.

[40] A. C. Guyton and J. E. Hall, *Textbook of medical physiology*, 11th ed., ser. Guyton Physiology. London, England: W B Saunders, Jul. 2005.

[41] K. S. Saladin, *Anatomy and Physiology*, 4th ed. Maidenhead, England: McGraw Hill Higher Education, Mar. 2006.

[42] W. Penfield and E. Boldrey, "Somatic motor and sensory representation in the cerebral cortex of man as studied by electrical stimulation," *Brain*, vol. 60, no. 4, pp. 389–443, 1937.

[43] E. V. Evarts, "Relation of pyramidal tract activity to force exerted during voluntary movement." *Journal of Neurophysiology*, vol. 31, no. 1, pp. 14–27, Jan. 1968.

[44] A. P. Georgopoulos, A. B. Schwartz, and R. E. Kettner, "Neuronal population coding of movement direction," *Science*, vol. 233, no. 4771, pp. 1416–1419, Sep. 1986.

[45] S. Kakei, D. S. Hoffman, and P. L. Strick, "Muscle and movement representations in the primary motor cortex," *Science*, vol. 285, no. 5436, pp. 2136–2139, Sep. 1999.

[46] D. W. Moran and A. B. Schwartz, "Motor cortical representation of speed and direction during reaching," *Journal of Neurophysiology*, vol. 82, no. 5, pp. 2676–2692, Nov. 1999.

[47] E. Todorov, "Direct cortical control of muscle activation in voluntary arm movements: a model," *Nature Neuroscience*, vol. 3, no. 4, pp. 391–398, Apr. 2000.

[48] W. Prinz, "Perception and action planning," *European journal of cognitive psychology*, vol. 9, no. 2, pp. 129–154, 1997.

[49] D. J. Serrien, L. H. Strens, A. Oliviero, and P. Brown, "Repetitive transcranial magnetic stimulation of the supplementary motor area (SMA) degrades bimanual movement control in humans," *Neuroscience Letters*, vol. 328, no. 2, pp. 89–92, Aug. 2002.

[50] I. Jenkins, D. Brooks, P. Nixon, R. Frackowiak, and R. Passingham, "Motor sequence learning: a study with positron emission tomography," *The Journal of Neuroscience*, vol. 14, no. 6, pp. 3775–3790, 1994.

[51] N. Picard, "Activation of the supplementary motor area (SMA) during performance of visually guided movements," *Cerebral Cortex*, vol. 13, no. 9, pp. 977–986, Sep. 2003.

[52] S. Zhang, J. S. Ide, and C. shan R. Li, "Resting-state functional connectivity of the medial superior frontal cortex," *Cerebral Cortex*, vol. 22, no. 1, pp. 99–111, 2011.

[53] F. Buchthal and H. Schmalbruch, "Motor unit of mammalian muscle," *Physiological Reviews*, vol. 60, no. 1, pp. 90–142, Jan. 1980.

[54] M. C. Brown, J. K. Jansen, and D. V. Essen, "Polyneuronal innervation of skeletal muscle in new-born rats and its elimination during maturation." *The Journal of Physiology*, vol. 261, no. 2, pp. 387–422, Oct. 1976. [Online]. Available: https://doi.org/10.1113/jphysiol.1976.sp011565

[55] E. Henneman, "Relation between size of neurons and their susceptibility to discharge," *Science*, vol. 126, no. 3287, pp. 1345–1347, 1957.

[56] I. B. Levitan and L. K. Kaczmarek, "Signaling in the brain," in *The Neuron*. Oxford University Press, 2015, pp. 3–22.

[57] A. Sandow, "Excitation-contraction coupling in muscular response," *Yale Journal of Biology and Medicine*, vol. 25, no. 3, pp. 176–201, Dec. 1952.

[58] J. Feher, "Chemical foundations of physiology," in *Quantitative Human Physiology*. Elsevier, 2012, pp. 32–42.

[59] J. V. Basmajian and C. J. De Luca, *Muscles Alive: Their Functions Revealed by Electromyography*, 5th ed. Baltimore: Williams & Wilkins, 1985.

[60] J. Malmivuo and R. Plonsey, *Bioelectromagnetism: Principles and Applications of Bioelectric and Biomagnetic Fields*. New York: Oxford University Press, 1995.

[61] S. D. Nandedkar, P. E. Barkhaus, and E. V. Stålberg, "Quantitative emg: Signal analysis and clinical applications," *Muscle & Nerve*, vol. 61, no. 1, pp. 28–41, 2020.

[62] H. Wu, G. Yang, K. Zhu, S. Liu, W. Guo, Z. Jiang, and Z. Li, "Materials, devices, and systems of on-skin electrodes for electrophysiological monitoring and human–machine interfaces," *Advanced Science*, vol. 8, no. 2, p. 2001938, Dec. 2020. [Online]. Available: https://doi.org/10.1002/advs.202001938

[63] J.-W. Jeong, M. K. Kim, H. Cheng, W.-H. Yeo, X. Huang, Y. Liu, Y. Zhang, Y. Huang, and J. A. Rogers, "Capacitive epidermal electronics for electrically safe, long-term electrophysiological measurements," *Advanced Healthcare Materials*, vol. 3, no. 5, pp. 642–648, Oct. 2013. [Online]. Available: https://doi.org/10.1002/adhm.201300334

[64] W. Gao, H. Ota, D. Kiriya, K. Takei, and A. Javey, "Flexible electronics toward wearable sensing," *Accounts of Chemical Research*, vol. 52, no. 3, pp. 523–533, Feb. 2019. [Online]. Available: https://doi.org/10.1021/acs.accounts.8b00500

[65] D. Farina, R. Merletti, and R. M. Enoka, "The extraction of neural strategies from the surface EMG," *Journal of Applied Physiology*, vol. 96, no. 4, pp. 1486–1495, Apr. 2004. [Online]. Available: https://doi.org/10.1152/japplphysiol.01070.2003

[66] G. Drost, D. F. Stegeman, B. G. van Engelen, and M. J. Zwarts, "Clinical applications of high-density surface EMG: A systematic review," *Journal of Electromyography and Kinesiology*, vol. 16, no. 6, pp. 586–602, Dec. 2006. [Online]. Available: https://doi.org/10.1016/j.jelekin.2006.09.005

[67] F. Negro, S. Muceli, A. M. Castronovo, A. Holobar, and D. Farina, "Multi-channel intramuscular and surface EMG decomposition by convolutive blind source separation," *Journal of Neural Engineering*, vol. 13, no. 2, p. 026027, Feb. 2016. [Online]. Available: https://doi.org/10.1088/1741-2560/13/2/026027

[68] A. Searle and L. Kirkup, "A direct comparison of wet, dry and insulating bioelectric recording electrodes," *Physiological Measurement*, vol. 21, no. 2, pp. 271–283, May 2000. [Online]. Available: https://doi.org/10.1088/0967-3334/21/2/307

[69] K. E. Mathewson, T. J. L. Harrison, and S. A. D. Kizuk, "High and dry? comparing active dry EEG electrodes to active and passive wet electrodes," *Psychophysiology*, vol. 54, no. 1, pp. 74–82, Dec. 2016. [Online]. Available: https://doi.org/10.1111/psyp.12536

[70] C. Grönlund, "Spatio-temporal processing of surface electromyographic signals: information on neuromuscular function and control," Ph.D. dissertation, Umeå universitet, 2006.

[71] D. I. Rubin, "Needle electromyography: Basic concepts," in *Clinical Neurophysiology: Basis and Technical Aspects*. Elsevier, 2019, pp. 243–256. [Online]. Available: https://doi.org/10.1016/b978-0-444-64032-1.00016-3

[72] J. Chae, J. Knutson, R. Hart, and Z.-P. Fang, "Selectivity and sensitivity of intramuscular and transcutaneous electromyography electrodes," *American Journal of Physical Medicine & Rehabilitation*, vol. 80, no. 5, pp. 374–379, May 2001. [Online]. Available: https://doi.org/10.1097/00002060-200105000-00010

[73] S. Muceli, K. D. Bergmeister, K.-P. Hoffmann, M. Aman, I. Vukajlija, O. C. Aszmann, and D. Farina, "Decoding motor neuron activity from epimysial thin-film electrode recordings following targeted muscle reinnervation," *Journal of Neural Engineering*, vol. 16, no. 1, p. 016010, Dec. 2018. [Online]. Available: https://doi.org/10.1088/1741-2552/aaed85

[74] OpenAI, "Gpt-4 technical report," 2023.

[75] T. Mitchell, *Machine Learning*, ser. McGraw-Hill series in computer science. New York, NY: McGraw-Hill Professional, Mar. 1997.

[76] K. P. Murphy, *Probabilistic Machine Learning: An introduction*. MIT Press, 2022. [Online]. Available: probml.ai

[77] S. Sra, S. Nowozin, and S. J. Wright, *Optimization for machine learning*. London, England: MIT Press, Sep. 2011.

[78] W. Mettrey, "An assessment of tools for building large knowledge-based systems," *AI magazine*, vol. 8, no. 4, pp. 81–81, 1987.

[79] Bruce, *What can be automated?*, ser. The MIT Press series in computer science. London, England: MIT Press, Jan. 1980.

[80] S. Russell and P. Norvig, *Artificial intelligence*, 3rd ed. Upper Saddle River, NJ: Pearson, Dec. 2009.

[81] R. Sutton, "The bitter lesson," *Incomplete Ideas (blog)*, vol. 13, no. 1, 2019.

[82] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, no. 8, pp. 114–117, 1965.

[83] C. Mead and M. Ismail, *Analog VLSI Implementation of Neural Systems*. Springer US, 1989. [Online]. Available: https://doi.org/10.1007/978-1-4613-1639-8

[84] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.

[85] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[86] S. B. Kotsiantis, I. Zaharakis, P. Pintelas *et al.*, "Supervised machine learning: A review of classification techniques," *Emerging artificial intelligence applications in computer engineering*, vol. 160, no. 1, pp. 3–24, 2007.

[87] G. Hinton and T. J. Sejnowski, Eds., *Unsupervised learning*, ser. Computational Neuroscience Series. Cambridge, MA: Bradford Books, May 1999.

[88] A. Saxena, M. Prasad, A. Gupta, N. Bharill, O. P. Patel, A. Tiwari, M. J. Er, W. Ding, and C.-T. Lin, "A review of clustering techniques and developments," *Neurocomputing*, vol. 267, pp. 664–681, Dec. 2017. [Online]. Available: https://doi.org/10.1016/j.neucom.2017.06.053

[89] J. Bovy, D. W. Hogg, and S. T. Roweis, "Extreme deconvolution: Inferring complete distribution functions from noisy, heterogeneous and incomplete observations," 2009. [Online]. Available: https://arxiv.org/abs/0905.2979

[90] T. White, "Sampling generative networks," 2016. [Online]. Available: https://arxiv.org/abs/1609.04468

[91] X. Zhu and A. B. Goldberg, "Introduction to semi-supervised learning," *Synthesis lectures on artificial intelligence and machine learning*, vol. 3, no. 1, pp. 1–130, 2009.

[92] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the eleventh annual conference on Computational learning theory*, 1998, pp. 92–100.

[93] J. Zhao, X. Xie, X. Xu, and S. Sun, "Multi-view learning overview: Recent progress and new challenges," *Information Fusion*, vol. 38, pp. 43–54, Nov. 2017. [Online]. Available: https://doi.org/10.1016/j.inffus.2017.02.007

[94] Z. Song, X. Yang, Z. Xu, and I. King, "Graph-based semi-supervised learning: A comprehensive review," 2021. [Online]. Available: https://arxiv.org/abs/2102.13303

[95] P. Bachman, O. Alsharif, and D. Precup, "Learning with pseudo-ensembles," *Advances in neural information processing systems*, vol. 27, 2014.

[96] C. Doersch and A. Zisserman, "Multi-task self-supervised visual learning," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2070–2079.

[97] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever *et al.*, "Improving language understanding by generative pre-training," 2018.

[98] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2536–2544.

[99] R. S. Sutton and A. G. Barto, *Reinforcement Learning*, ser. Adaptive Computation and Machine Learning series. Cambridge, MA: Bradford Books, Feb. 1998.

[100] G. A. Rummery and M. Niranjan, *On-line Q-learning using connectionist systems*. University of Cambridge, Department of Engineering Cambridge, UK, 1994, vol. 37.

[101] S.-I. Amari, "Natural gradient works efficiently in learning," *Neural computation*, vol. 10, no. 2, pp. 251–276, 1998.

[102] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, no. 3-4, pp. 229–256, May 1992. [Online]. Available: https://doi.org/10.1007/bf00992696

[103] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," *Machine Learning*, vol. 85, no. 3, pp. 333–359, Jun. 2011. [Online]. Available: https://doi.org/10.1007/s10994-011-5256-5

[104] E. Fix and J. L. Hodges, "Discriminatory analysis, nonparametric discrimination: Consistency properties," USAF School of Aviation Medicine, Randolph Field, Texas, Technical Report 21-49-004, 1951.

[105] M.-L. Zhang and Z.-H. Zhou, "Ml-knn: A lazy learning approach to multi-label learning," *Pattern recognition*, vol. 40, no. 7, pp. 2038–2048, 2007.

[106] K. Chomboon, P. Chujai, P. Teerarassamee, K. Kerdprasop, and N. Kerdprasop, "An empirical study of distance metrics for k-nearest neighbor algorithm," in *Proceedings of the 3rd international conference on industrial application engineering*, vol. 2, 2015.

[107] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[108] R. O. Duda, P. E. Hart *et al.*, *Pattern classification*. John Wiley & Sons, 2006.

[109] H. Wang, C. Ding, and H. Huang, "Multi-label linear discriminant analysis," in *Computer Vision – ECCV 2010*. Springer Berlin Heidelberg, 2010, pp. 126–139. [Online]. Available: https://doi.org/10.1007/978-3-642-15567-3_10

[110] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[111] A. Aizerman, "Theoretical foundations of the potential function method in pattern recognition learning," *Automation and remote control*, vol. 25, pp. 821–837, 1964.

[112] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*. ACM, Jul. 1992. [Online]. Available: https://doi.org/10.1145/130385.130401

[113] K.-B. Duan and S. S. Keerthi, "Which is the best multiclass SVM method? an empirical study," in *Multiple Classifier Systems*. Springer Berlin Heidelberg, 2005, pp. 278–285. [Online]. Available: https://doi.org/10.1007/11494683_28

[114] D. A. Freedman, *Statistical models: theory and practice*. cambridge university press, 2009.

[115] H. Goldstein, "Multilevel mixed linear model analysis using iterative generalized least squares," *Biometrika*, vol. 73, no. 1, pp. 43–56, 1986. [Online]. Available: https://doi.org/10.1093/biomet/73.1.43

[116] R. M. Bethea, B. S. Duran, and T. L. Boullion, *Statistical methods for engineers and scientists*, 2nd ed., ser. Statistics: Textbooks & Monographs. New York, NY: Marcel Dekker, Jan. 1985.

[117] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quarterly of Applied Mathematics*, vol. 2, no. 2, pp. 164–168, 1944. [Online]. Available: https://doi.org/10.1090/qam/10666

[118] L.-C. Böiers, *Mathematical methods of optimization*. Studentlitteratur AB, 2010.

[119] B. Boehmke and B. M. Greenwell, *Hands-on machine learning with R*, ser. Chapman & Hall/CRC The R Series. London, England: CRC Press, 2019.

[120] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosophical Magazine*, vol. 2, no. 11, pp. 559–572, 1901.

[121] D. Hsu, S. M. Kakade, and T. Zhang, "A spectral algorithm for learning hidden markov models," 2008. [Online]. Available: https://arxiv.org/abs/0811.4413

[122] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, Oct. 1999.

[123] D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," *Advances in neural information processing systems*, vol. 13, 2000.

[124] D. H. Wolpert, W. G. Macready *et al.*, "No free lunch theorems for search," Citeseer, Tech. Rep., 1995.

[125] J. Baxter, "A model of inductive bias learning," *Journal of Artificial Intelligence Research*, vol. 12, pp. 149–198, Mar. 2000. [Online]. Available: https://doi.org/10.1613/jair.731

[126] A. Goyal and Y. Bengio, "Inductive biases for deep learning of higher-level cognition," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 478, no. 2266, Oct. 2022. [Online]. Available: https://doi.org/10.1098/rspa.2021.0068

[127] S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias/variance dilemma," *Neural Computation*, vol. 4, no. 1, pp. 1–58, Jan. 1992. [Online]. Available: https://doi.org/10.1162/neco.1992.4.1.1

[128] M. Stone, "Cross-validatory choice and assessment of statistical predictions," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 36, no. 2, pp. 111–133, Jan. 1974. [Online]. Available: https://doi.org/10.1111/j.2517-6161.1974.tb00994.x

[129] A. M. Molinaro, R. Simon, and R. M. Pfeiffer, "Prediction error estimation: a comparison of resampling methods," *Bioinformatics*, vol. 21, no. 15, pp. 3301–3307, May 2005. [Online]. Available: https://doi.org/10.1093/bioinformatics/bti499

[130] B. Efron, "Bootstrap methods: Another look at the jackknife," *The Annals of Statistics*, vol. 7, no. 1, Jan. 1979. [Online]. Available: https://doi.org/10.1214/aos/1176344552

[131] S. V. Stehman, "Selecting and interpreting measures of thematic classification accuracy," *Remote Sensing of Environment*, vol. 62, no. 1, pp. 77–89, Oct. 1997. [Online]. Available: https://doi.org/10.1016/s0034-4257(97)00083-7

[132] D. M. W. Powers, "Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation," *ArXiv*, vol. abs/2010.16061, 2011.

[133] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *International Journal of Forecasting*, vol. 22, no. 4, pp. 679–688, Oct. 2006. [Online]. Available: https://doi.org/10.1016/j.ijforecast.2006.03.001

[134] D. K. Luu, A. T. Nguyen, M. Jiang, J. Xu, M. W. Drealan, J. Cheng, E. W. Keefer, Q. Zhao, and Z. Yang, "Deep learning-based approaches for decoding motor intent from peripheral nerve signals," *Frontiers in Neuroscience*, vol. 15, Jun. 2021. [Online]. Available: https://doi.org/10.3389/fnins.2021.667907

[135] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[136] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[137] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal dynamics*. Cambridge, England: Cambridge University Press, 2014.

[138] J. Schmidhuber, "Annotated history of modern ai and deep learning," 2022. [Online]. Available: https://arxiv.org/abs/2212.11279

[139] M. F. Abbod, J. W. Catto, D. A. Linkens, and F. C. Hamdy, "Application of artificial intelligence to the management of urological cancer," *Journal of Urology*, vol. 178, no. 4, pp. 1150–1156, Oct. 2007. [Online]. Available: https://doi.org/10.1016/j.juro.2007.05.122

[140] G. Hinton, L. Deng, D. Yu, G. Dahl, A. rahman Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, Nov. 2012. [Online]. Available: https://doi.org/10.1109/msp.2012.2205597

[141] K. Fukushima, "Cognitron: A self-organizing multilayered neural network," *Biological Cybernetics*, vol. 20, no. 3-4, pp. 121–136, 1975. [Online]. Available: https://doi.org/10.1007/bf00342633

[142] A. K. Bhoi, P. K. Mallick, C.-M. Liu, and V. E. Balas, Eds., *Bio-inspired Neurocomputing*. Springer Singapore, 2021. [Online]. Available: https://doi.org/10.1007/978-981-15-5495-7

[143] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," 2015. [Online]. Available: https://arxiv.org/abs/1511.07289

[144] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," 2016. [Online]. Available: https://arxiv.org/abs/1606.08415

[145] A. Maas, A. Hannun, and A. Ng, "Rectifier nonlinearities improve neural network acoustic models," 2013.

[146] S. Sutherland, "Eye, brain and vision," *Nature*, vol. 362, no. 6419, pp. 419–420, 1993. [Online]. Available: https://doi.org/10.1038/362419a0

[147] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, Dec. 1943. [Online]. Available: https://doi.org/10.1007/bf02478259

[148] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," 2017. [Online]. Available: https://arxiv.org/abs/1706.04599

[149] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[150] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, no. 2, pp. 251–257, 1991. [Online]. Available: https://doi.org/10.1016/0893-6080(91)90009-t

[151] B. van Merrienboer, O. Breuleux, A. Bergeron, and P. Lamblin, "Automatic differentiation in ml: Where we are and where we should be going," in *Advances in Neural Information Processing Systems*, vol. 31, 2018. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2018/file/770f8e448d07586afbf77bb59f698587-Paper.pdf

[152] M. Tegmark, *Life 3.0*. Knopf Publishing Group, Aug. 2017.

[153] J. Kossen, N. Band, C. Lyle, A. N. Gomez, T. Rainforth, and Y. Gal, "Self-attention between datapoints: Going beyond individual input-output pairs in deep learning," 2021. [Online]. Available: https://arxiv.org/abs/2106.02584

[154] S. Kullback and R. A. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, Mar. 1951. [Online]. Available: https://doi.org/10.1214/aoms/1177729694

[155] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural networks: Tricks of the trade*. Springer, 1998, pp. 9–50.

[156] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[157] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[158] K. Wong, R. Dornberger, and T. Hanne, "An analysis of weight initialization methods in connection with different activation functions for feedforward neural networks," *Evolutionary Intelligence*, Nov. 2022. [Online]. Available: https://doi.org/10.1007/s12065-022-00795-y

[159] A. G. Ivakhnenko, V. G. Lapa, and V. G. Lapa, *Cybernetics and forecasting techniques*. American Elsevier Publishing Company, 1967, vol. 8.

[160] T. Bäck and H.-P. Schwefel, "An overview of evolutionary algorithms for parameter optimization," *Evolutionary computation*, vol. 1, no. 1, pp. 1–23, 1993.

[161] E. Real, S. Moore, A. Selle, S. Saxena, Y. L. Suematsu, J. Tan, Q. V. Le, and A. Kurakin, "Large-scale evolution of image classifiers," in *International Conference on Machine Learning*. PMLR, 2017, pp. 2902–2911.

[162] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, May 1983. [Online]. Available: https://doi.org/10.1126/science.220.4598.671

[163] L. R. Rere, M. I. Fanany, and A. M. Arymurthy, "Simulated annealing algorithm for deep learning," *Procedia Computer Science*, vol. 72, pp. 137–144, 2015. [Online]. Available: https://doi.org/10.1016/j.procs.2015.12.114

[164] A. J. Shepherd, *Second-order methods for neural networks: Fast and reliable training methods for multi-layer perceptrons*. Springer Science & Business Media, 2012.

[165] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[166] J. Hadamard, *Mémoire sur le problème d'analyse relatif à l'équilibre des plaques élastiques encastrées*. Imprimerie nationale, 1908, vol. 33.

[167] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[168] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, Apr. 1980. [Online]. Available: https://doi.org/10.1007/bf00344251

[169] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[170] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, May 2017. [Online]. Available: https://doi.org/10.1145/3065386

[171] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014. [Online]. Available: https://arxiv.org/abs/1409.1556

[172] D. A. Forsyth, J. L. Mundy, V. di Gesú, R. Cipolla, Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio, "Object recognition with gradient-based learning," *Shape, contour and grouping in computer vision*, pp. 319–345, 1999.

[173] Y. LeCun, F. J. Huang, and L. Bottou, "Learning methods for generic object recognition with invariance to pose and lighting," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 2, 2004, pp. II–104 Vol.2.

[174] Y.-T. Zhou and R. Chellappa, "Computation of optical flow using a neural network." in *ICNN*, 1988, pp. 71–78.

[175] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," *Advances in neural information processing systems*, vol. 30, 2017.

[176] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," *arXiv preprint arXiv:1412.6806*, 2014.

[177] M. Lin, Q. Chen, and S. Yan, "Network in network," 2013. [Online]. Available: https://arxiv.org/abs/1312.4400

[178] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.

[179] S. Basodi, C. Ji, H. Zhang, and Y. Pan, "Gradient amplification: An efficient way to train deep neural networks," *Big Data Mining and Analytics*, vol. 3, no. 3, pp. 196–207, Sep. 2020. [Online]. Available: https://doi.org/10.26599/bdma.2020.9020004

[180] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[181] F. A. Gers, J. Schmidhuber, and F. Cummins, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proceedings of the Seventh International Conference on Spoken Language Processing (ICSLP2000)*. ISCA, 2000.

[182] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," 2014. [Online]. Available: https://arxiv.org/abs/1406.1078

[183] S. Yang, X. Yu, and Y. Zhou, "Lstm and gru neural network performance comparison study: Taking yelp review dataset as an example," in *2020 International workshop on electronic communication and artificial intelligence (IWECAI)*. IEEE, 2020, pp. 98–101.

[184] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[185] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[186] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 2020. [Online]. Available: https://arxiv.org/abs/2010.11929

[187] D. Rothman and A. Gulli, *Transformers for Natural Language Processing*. Packt Publishing Ltd, 2022.

[188] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2018. [Online]. Available: https://arxiv.org/abs/1810.04805

[189] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," 2022. [Online]. Available: https://arxiv.org/abs/2212.04356

[190] K. Choromanski, V. Likhosherstov, D. Dohan, X. Song, A. Gane, T. Sarlos, P. Hawkins, J. Davis, A. Mohiuddin, L. Kaiser, D. Belanger, L. Colwell, and A. Weller, "Rethinking attention with performers," 2020. [Online]. Available: https://arxiv.org/abs/2009.14794

[191] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," 2020. [Online]. Available: https://arxiv.org/abs/2004.05150

[192] A. Jaegle, F. Gimeno, A. Brock, A. Zisserman, O. Vinyals, and J. Carreira, "Perceiver: General perception with iterative attention," 2021. [Online]. Available: https://arxiv.org/abs/2103.03206

[193] T. Lin, Y. Wang, X. Liu, and X. Qiu, "A survey of transformers," 2021. [Online]. Available: https://arxiv.org/abs/2106.04554

[194] J. Schaffer, "What not to multiply without necessity," *Australasian Journal of Philosophy*, vol. 93, no. 4, pp. 644–664, Dec. 2014. [Online]. Available: https://doi.org/10.1080/00048402.2014.992447

[195] A. Kolmogorov, "On tables of random numbers," *Theoretical Computer Science*, vol. 207, no. 2, pp. 387–395, Nov. 1998. [Online]. Available: https://doi.org/10.1016/s0304-3975(98)00075-9

[196] K. C. Green and J. S. Armstrong, "Simple versus complex forecasting: The evidence," *Journal of Business Research*, vol. 68, no. 8, pp. 1678–1685, Aug. 2015. [Online]. Available: https://doi.org/10.1016/j.jbusres.2015.03.026

[197] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[198] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*. PMLR, 2016, pp. 1050–1059.

[199] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, Jul. 2019. [Online]. Available: https://doi.org/10.1186/s40537-019-0197-0

[200] P. Tsinganos, B. Cornelis, J. Cornelis, B. Jansen, and A. Skodras, "Data augmentation of surface electromyography for hand gesture recognition," *Sensors*, vol. 20, no. 17, p. 4892, Aug. 2020. [Online]. Available: https://doi.org/10.3390/s20174892

[201] I. O. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, A. Steiner, D. Keysers, J. Uszkoreit *et al.*, "Mlp-mixer: An all-mlp architecture for vision," *Advances in neural information processing systems*, vol. 34, pp. 24 261–24 272, 2021.

[202] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[203] A. P. Badia, B. Piot, S. Kapturowski, P. Sprechmann, A. Vitvitskyi, D. Guo, and C. Blundell, "Agent57: Outperforming the atari human benchmark," 2020. [Online]. Available: https://arxiv.org/abs/2003.13350

[204] M. Hutson, "Has artificial intelligence become alchemy?" *Science*, vol. 360, no. 6388, pp. 478–478, 2018. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.360.6388.478

[205] N. Cohen, O. Sharir, and A. Shashua, "On the expressive power of deep learning: A tensor analysis," in *Conference on Learning Theory*, 2016, pp. 698–728.

[206] G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio, "On the number of linear regions of deep neural networks," *Advances in neural information processing systems*, vol. 27, 2014.

[207] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*. Springer, 2014, pp. 818–833.

[208] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *SIAM Review*, vol. 60, no. 2, pp. 223–311, 2018.

[209] A. R. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Transactions on Information theory*, vol. 39, no. 3, pp. 930–945, 1993.

[210] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On large-batch training for deep learning: Generalization gap and sharp minima," *arXiv preprint arXiv:1609.04836*, 2016.

[211] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. G. Wilson, "Averaging weights leads to wider optima and better generalization," *arXiv preprint arXiv:1803.05407*, 2018.

[212] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," *arXiv preprint arXiv:1611.03530*, 2016.

[213] M. Belkin, D. Hsu, S. Ma, and S. Mandal, "Reconciling modern machine-learning practice and the classical bias–variance trade-off," *Proceedings of the National Academy of Sciences*, vol. 116, no. 32, pp. 15 849–15 854, 2019.

[214] M. Belkin, D. Hsu, and J. Xu, "Two models of double descent for weak features," *SIAM Journal on Mathematics of Data Science*, vol. 2, no. 4, pp. 1167–1180, 2020.

[215] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," *arXiv preprint arXiv:1803.03635*, 2018.

[216] A. Morcos, H. Yu, M. Paganini, and Y. Tian, "One ticket to win them all: generalizing lottery ticket initializations across datasets and optimizers," *Advances in neural information processing systems*, vol. 32, 2019.

[217] N. Wiener, *Cybernetics*, 2nd ed., ser. The MIT Press. London, England: MIT Press, Jan. 1961.

[218] A. Chowdhury, R. Ramadas, and S. Karmakar, "Muscle computer interface: A review," in *Lecture Notes in Mechanical Engineering*. Springer India, 2013, pp. 411–421. [Online]. Available: https://doi.org/10.1007/978-81-322-1050-4_33

[219] P. Parker, K. Englehart, and B. Hudgins, "Myoelectric signal processing for control of powered limb prostheses," *Journal of Electromyography and Kinesiology*, vol. 16, no. 6, pp. 541–548, Dec. 2006. [Online]. Available: https://doi.org/10.1016/j.jelekin.2006.08.006

[220] O. M. Giggins, U. Persson, and B. Caulfield, "Biofeedback in rehabilitation," *Journal of NeuroEngineering and Rehabilitation*, vol. 10, no. 1, p. 60, 2013. [Online]. Available: https://doi.org/10.1186/1743-0003-10-60

[221] J. L. Contreras-Vidal, N. A. Bhagat, J. Brantley, J. G. Cruz-Garza, Y. He, Q. Manley, S. Nakagome, K. Nathan, S. H. Tan, F. Zhu *et al.*, "Powered exoskeletons for bipedal locomotion after spinal cord injury," *Journal of neural engineering*, vol. 13, no. 3, p. 031001, 2016.

[222] R. H. Chowdhury, M. Reaz, A. Bakar, and M. S. Hasan, "Muscle contraction: The subtle way of human computer interaction," *Research Journal of Applied Sciences, Engineering and Technology*, vol. 6, no. 12, pp. 2192–2196, Jul. 2013.

[223] J. M. Hahne, M. Markovic, and D. Farina, "User adaptation in myoelectric man-machine interfaces," *Scientific reports*, vol. 7, no. 1, p. 4437, 2017.

[224] M. Zecca, S. Micera, M. C. Carrozza, and P. Dario, "Control of multifunctional prosthetic hands by processing the electromyographic signal," *Critical Reviews™ in Biomedical Engineering*, vol. 30, no. 4-6, 2002.

[225] D. Farina, N. Jiang, H. Rehbaum, A. Holobar, B. Graimann, H. Dietl, and O. C. Aszmann, "The extraction of neural information from the surface EMG for the control of upper-limb prostheses: Emerging avenues and challenges," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, no. 4, pp. 797–809, Jul. 2014. [Online]. Available: https://doi.org/10.1109/tnsre.2014.2305111

[226] M. B. I. Reaz, M. S. Hussain, and F. Mohd-Yasin, "Techniques of emg signal analysis: detection, processing, classification and applications," *Biological procedures online*, vol. 8, pp. 11–35, 2006.

[227] K. Englehart and B. Hudgins, "A robust, real-time control scheme for multifunction myoelectric control," *IEEE transactions on biomedical engineering*, vol. 50, no. 7, pp. 848–854, 2003.

[228] F. Sebelius, L. Eriksson, C. Balkenius, and T. Laurell, "Myoelectric control of a computer animated hand: A new concept based on the combined use of a tree-structured artificial neural network and a data glove," *Journal of medical engineering & technology*, vol. 30, no. 1, pp. 2–10, 2006.

[229] M. Ortiz-Catalan, B. Håkansson, and R. Brånemark, "Real-time and simultaneous control of artificial limbs based on pattern recognition algorithms," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, no. 4, pp. 756–764, 2014.

[230] A. J. Young, L. H. Smith, E. J. Rouse, and L. J. Hargrove, "Classification of simultaneous movements using surface emg pattern recognition," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 5, pp. 1250–1258, 2012.

[231] H. Rehbaum, N. Jiang, L. Paredes, S. Amsuess, B. Graimann, and D. Farina, "Real time simultaneous and proportional control of multiple degrees of freedom from surface emg: preliminary results on subjects with limb deficiency," in *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 2012, pp. 1346–1349.

[232] R. Merletti and G. Cerone, "Tutorial. surface EMG detection, conditioning and pre-processing: Best practices," *Journal of Electromyography and Kinesiology*, vol. 54, p. 102440, Oct. 2020. [Online]. Available: https://doi.org/10.1016/j.jelekin.2020.102440

[233] V. Gupta, N. P. Reddy, and E. P. Canilang, "Surface emg measurements at the throat during dry and wet swallowing," *Dysphagia*, vol. 11, pp. 173–179, 1996.

[234] H. Nyquist, "Certain topics in telegraph transmission theory," *Transactions of the American Institute of Electrical Engineers*, vol. 47, no. 2, pp. 617–644, Apr. 1928. [Online]. Available: https://doi.org/10.1109/t-aiee.1928.5055024

[235] R. Merletti and P. Di Torino, "Standards for reporting emg data," *Journal of Electromyography and Kinesiology*, vol. 9, no. 1, pp. 3–4, 1999.

[236] L. Sörnmo and P. Laguna, *Bioelectrical signal processing in cardiac and neurological applications*. Academic press, 2005, vol. 8.

[237] M. Zivanovic and M. González-Izal, "Simultaneous powerline interference and baseline wander removal from ECG and EMG signals by sinusoidal modeling," *Medical Engineering & Physics*, vol. 35, no. 10, pp. 1431–1441, Oct. 2013. [Online]. Available: https://doi.org/10.1016/j.medengphy.2013.03.015

[238] C. De Luca, "Electromyography," *Encyclopedia of medical devices and instrumentation*, 2006.

[239] A. E. Olsson, N. Malešević, A. Björkman, and C. Antfolk, "Learning regularized representations of categorically labelled surface EMG enables simultaneous and proportional myoelectric control," *Journal of NeuroEngineering and Rehabilitation*, vol. 18, no. 1, Feb. 2021. [Online]. Available: https://doi.org/10.1186/s12984-021-00832-4

[240] L. H. Smith, L. J. Hargrove, B. A. Lock, and T. A. Kuiken, "Determining the optimal window length for pattern recognition-based myoelectric control: Balancing the competing effects of classification error and controller delay," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 19, no. 2, pp. 186–192, Apr. 2011. [Online]. Available: https://doi.org/10.1109/tnsre.2010.2100828

[241] R. N. Khushaba and K. Nazarpour, "Decoding HD-EMG signals for myoelectric control - how small can the analysis window size be?" *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 8569–8574, Oct. 2021. [Online]. Available: https://doi.org/10.1109/lra.2021.3111850

[242] A. Phinyomark, S. Thongpanja, H. Hu, P. Phukpattaranont, and C. Limsakul, "The usefulness of mean and median frequencies in electromyography analysis," *Computational intelligence in electromyography analysis-A perspective on current applications and future challenges*, vol. 81, p. 67, 2012.

[243] A. Phinyomark, P. Phukpattaranont, and C. Limsakul, "Feature reduction and selection for emg signal classification," *Expert systems with applications*, vol. 39, no. 8, pp. 7420–7431, 2012.

[244] A. Phinyomark, R. N. Khushaba, E. Ibánez-Marcelo, A. Patania, E. Scheme, and G. Petri, "Navigating features: a topologically informed chart of electromyographic features space," *Journal of The Royal Society Interface*, vol. 14, no. 137, p. 20170734, 2017.

[245] U. Côté-Allard, E. Campbell, A. Phinyomark, F. Laviolette, B. Gosselin, and E. Scheme, "Interpreting deep learning features for myoelectric control: A comparison with handcrafted features," *Frontiers in Bioengineering and Biotechnology*, vol. 8, Mar. 2020. [Online]. Available: https://doi.org/10.3389/fbioe.2020.00158

[246] A. Fougner, E. Scheme, A. D. C. Chan, K. Englehart, and Ø. Stavdahl, "Resolving the limb position effect in myoelectric pattern recognition," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 19, no. 6, pp. 644–651, Dec. 2011. [Online]. Available: https://doi.org/10.1109/tnsre.2011.2163529

[247] M. A. Oskoei and H. Hu, "Support vector machine-based classification scheme for myoelectric control applied to upper limb," *IEEE transactions on biomedical engineering*, vol. 55, no. 8, pp. 1956–1965, 2008.

[248] M. Hakonen, H. Piitulainen, and A. Visala, "Current state of digital signal processing in myoelectric interfaces and related applications," *Biomedical Signal Processing and Control*, vol. 18, pp. 334–359, 2015.

[249] G. Shuman, Z. Durić, D. Barbará, J. Lin, and L. H. Gerber, "Improving the recognition of grips and movements of the hand using myoelectric signals," *BMC Medical Informatics and Decision Making*, vol. 16, pp. 65–83, 2016.

[250] N. Malešević, D. Marković, G. Kanitz, M. Controzzi, C. Cipriani, and C. Antfolk, "Vector autoregressive hierarchical hidden markov models for extracting finger movements using multichannel surface EMG signals," *Complexity*, vol. 2018, pp. 1–12, 2018. [Online]. Available: https://doi.org/10.1155/2018/9728264

[251] K. Englehart, B. Hudgin, and P. A. Parker, "A wavelet-based continuous classification scheme for multifunction myoelectric control," *IEEE Transactions on Biomedical Engineering*, vol. 48, no. 3, pp. 302–311, 2001.

[252] P. Kaufmann, K. Englehart, and M. Platzner, "Fluctuating emg signals: Investigating long-term effects of pattern matching algorithms," in *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*. IEEE, 2010, pp. 6357–6360.

[253] L. J. Hargrove, E. J. Scheme, K. B. Englehart, and B. S. Hudgins, "Multiple binary classifications via linear discriminant analysis for improved controllability of a powered prosthesis," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 18, no. 1, pp. 49–57, Feb. 2010. [Online]. Available: https://doi.org/10.1109/tnsre.2009.2039590

[254] M. Atzori, M. Cognolato, and H. Müller, "Deep learning with convolutional neural networks applied to electromyography data: A resource for the classification of movements for prosthetic hands," *Frontiers in neurorobotics*, vol. 10, p. 9, 2016.

[255] Z. Ding, C. Yang, Z. Tian, C. Yi, Y. Fu, and F. Jiang, "sEMG-based gesture recognition with convolution neural networks," *Sustainability*, vol. 10, no. 6, p. 1865, Jun. 2018. [Online]. Available: https://doi.org/10.3390/su10061865

[256] U. Côté-Allard, C. L. Fall, A. Drouin, A. Campeau-Lecours, C. Gosselin, K. Glette, F. Laviolette, and B. Gosselin, "Deep learning for electromyographic hand gesture signal classification using transfer learning," *IEEE transactions on neural systems and rehabilitation engineering*, vol. 27, no. 4, pp. 760–771, 2019.

[257] H. Chen, Y. Zhang, G. Li, Y. Fang, and H. Liu, "Surface electromyography feature extraction via convolutional neural network," *International Journal of Machine Learning and Cybernetics*, vol. 11, no. 1, pp. 185–196, May 2019. [Online]. Available: https://doi.org/10.1007/s13042-019-00966-x

[258] A. E. Olsson, A. Björkman, and C. Antfolk, "Automatic discovery of resource-restricted convolutional neural network topologies for myoelectric pattern recognition," *Computers in Biology and Medicine*, vol. 120, p. 103723, 2020.

[259] W. Geng, Y. Du, W. Jin, W. Wei, Y. Hu, and J. Li, "Gesture recognition by instantaneous surface EMG images," *Scientific Reports*, vol. 6, no. 1, Nov. 2016. [Online]. Available: https://doi.org/10.1038/srep36571

[260] A. E. Olsson, P. Sager, E. Andersson, A. Björkman, N. Malešević, and C. Antfolk, "Extraction of multi-labelled movement information from the raw hd-semg image with time-domain depth," *Scientific reports*, vol. 9, no. 1, pp. 1–10, 2019.

[261] N. Nasri, S. Orts-Escolano, F. Gomez-Donoso, and M. Cazorla, "Inferring static hand poses from a low-cost non-intrusive semg sensor," *Sensors*, vol. 19, no. 2, p. 371, 2019.

[262] A. Olsson, N. Malešević, A. Björkman, and C. Antfolk, "Exploiting the intertemporal structure of the upper-limb semg: Comparisons between an lstm network and cross-sectional myoelectric pattern recognition methods," in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2019, pp. 6611–6615.

[263] P. Koch, H. Phan, M. Maass, F. Katzberg, R. Mazur, and A. Mertins, "Recurrent neural networks with weighting loss for early prediction of hand movements," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 1152–1156.

[264] A. Samadani, "Gated recurrent neural networks for emg-based hand gesture classification. a comparative study," in *2018 40th annual international conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2018, pp. 1–4.

[265] Y. Hu, Y. Wong, W. Wei, Y. Du, M. Kankanhalli, and W. Geng, "A novel attention-based hybrid CNN-RNN architecture for sEMG-based gesture recognition," *PLOS ONE*, vol. 13, no. 10, p. e0206049, Oct. 2018. [Online]. Available: https://doi.org/10.1371/journal.pone.0206049

[266] M. Alfaro-Ponce and I. Chairez, "Continuous and recurrent pattern dynamic neural networks recognition of electrophysiological signals," *Biomedical Signal Processing and Control*, vol. 57, p. 101783, 2020.

[267] S. Zabihi, E. Rahimian, A. Asif, and A. Mohammadi, "Trahgr: Transformer for hand gesture recognition via electromyography," 2022. [Online]. Available: https://arxiv.org/abs/2203.16336

[268] E. Rahimian, S. Zabihi, A. Asif, D. Farina, S. F. Atashzar, and A. Mohammadi, "Temgnet: Deep transformer-based decoding of upperlimb semg for hand gestures recognition," 2021. [Online]. Available: https://arxiv.org/abs/2109.12379

[269] W. Zhang, T. Zhao, J. Zhang, and Y. Wang, "LST-EMG-net: Long short-term transformer feature fusion network for sEMG gesture recognition," *Frontiers in Neurorobotics*, vol. 17, Feb. 2023. [Online]. Available: https://doi.org/10.3389/fnbot.2023.1127338

[270] I. Sosin, D. Kudenko, and A. Shpilman, "Continuous gesture recognition from semg sensor data with recurrent neural networks and adversarial domain adaptation," in *2018 15Th international conference on control, automation, robotics and vision (ICARCV)*. IEEE, 2018, pp. 1436–1441.

[271] N. Malešević, A. Björkman, G. S. Andersson, A. Matran-Fernandez, L. Citi, C. Cipriani, and C. Antfolk, "A database of multi-channel intramuscular electromyogram signals during isometric hand muscles contractions," *Scientific Data*, vol. 7, no. 1, Jan. 2020. [Online]. Available: https://doi.org/10.1038/s41597-019-0335-8

[272] W. Yang, D. Yang, J. Li, Y. Liu, and H. Liu, "Emg dataset augmentation approaches for improving the multi-dof wrist movement regression accuracy and robustness," in *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2018, pp. 1268–1273.

[273] M. Atzori, A. Gijsberts, C. Castellini, B. Caputo, A.-G. M. Hager, S. Elsig, G. Giatsidis, F. Bassetto, and H. Müller, "Electromyography data for non-invasive naturally-controlled robotic hand prostheses," *Scientific Data*, vol. 1, no. 1, Dec. 2014. [Online]. Available: https://doi.org/10.1038/sdata.2014.53

[274] A. L. Rincon, H. Yamasaki, and S. Shimoda, "Design of a video game for rehabilitation using motion capture, EMG analysis and virtual reality," in *2016 International Conference on Electronics, Communications and Computers (CONIELECOMP)*. IEEE, Feb. 2016. [Online]. Available: https://doi.org/10.1109/conielecomp.2016.7438575

[275] M. Cognolato, A. Gijsberts, V. Gregori, G. Saetta, K. Giacomino, A.-G. M. Hager, A. Gigli, D. Faccio, C. Tiengo, F. Bassetto, B. Caputo, P. Brugger, M. Atzori, and H. Müller, "Gaze, visual, myoelectric, and inertial data of grasps for intelligent prosthetics," *Scientific Data*, vol. 7, no. 1, Feb. 2020. [Online]. Available: https://doi.org/10.1038/s41597-020-0380-3

[276] N. Malešević, A. Olsson, P. Sager, E. Andersson, C. Cipriani, M. Controzzi, A. Björkman, and C. Antfolk, "A database of high-density surface electromyogram signals comprising 65 isometric hand gestures," *Scientific Data*, vol. 8, no. 1, Feb. 2021. [Online]. Available: https://doi.org/10.1038/s41597-021-00843-9

[277] E. A. Clancy, L. Liu, P. Liu, and D. V. Z. Moyer, "Identification of constant-posture emg–torque relationship about the elbow using nonlinear dynamic

models," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 1, pp. 205–212, 2011.

[278] Q. Ding, J. Han, and X. Zhao, "Continuous estimation of human multi-joint angles from semg using a state-space model," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 9, pp. 1518–1528, 2016.

[279] W. Meng, B. Ding, Z. Zhou, Q. Liu, and Q. Ai, "An emg-based force prediction and control approach for robot-assisted lower limb rehabilitation," in *2014 IEEE international conference on systems, man, and cybernetics (SMC)*. IEEE, 2014, pp. 2198–2203.

[280] J.-L. Ren, Y.-H. Chien, E.-Y. Chia, L.-C. Fu, and J.-S. Lai, "Deep learning based motion prediction for exoskeleton robot control in upper limb rehabilitation," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 5076–5082.

[281] A. Ameri, M. A. Akhaee, E. Scheme, and K. Englehart, "Regression convolutional neural network for improved simultaneous emg control," *Journal of neural engineering*, vol. 16, no. 3, p. 036015, 2019.

[282] A. E. Olsson, N. Malešević, A. Björkman, and C. Antfolk, "End-to-end estimation of hand-and wrist forces from raw intramuscular emg signals using lstm networks," *Frontiers in Neuroscience*, p. 1555, 2021.

[283] J. L. Nielsen, S. Holmgaard, N. Jiang, K. B. Englehart, D. Farina, and P. A. Parker, "Simultaneous and proportional force estimation for multifunction myoelectric prostheses using mirrored bilateral training," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 3, pp. 681–688, 2010.

[284] N. Jiang, H. Rehbaum, I. Vujaklija, B. Graimann, and D. Farina, "Intuitive, online, simultaneous, and proportional myoelectric control over two degrees-of-freedom in upper limb amputees," *IEEE transactions on neural systems and rehabilitation engineering*, vol. 22, no. 3, pp. 501–510, 2013.

[285] E. N. Kamavuako, E. J. Scheme, and K. B. Englehart, "On the usability of intramuscular emg for prosthetic control: A fitts' law approach," *Journal of Electromyography and Kinesiology*, vol. 24, no. 5, pp. 770–777, 2014.

[286] E. Scheme, B. Lock, L. Hargrove, W. Hill, U. Kuruganti, and K. Englehart, "Motion normalized proportional control for improved pattern recognition-based

myoelectric control," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, no. 1, pp. 149–157, 2013.

[287] A. Holobar, M. A. Minetto, A. Botter, F. Negro, and D. Farina, "Experimental analysis of accuracy in the identification of motor unit spike trains from high-density surface EMG," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 18, no. 3, pp. 221–229, Jun. 2010. [Online]. Available: https://doi.org/10.1109/tnsre.2010.2041593

[288] A. Holobar and D. Zazula, "Multichannel blind source separation using convolution kernel compensation," *IEEE Transactions on Signal Processing*, vol. 55, no. 9, pp. 4487–4496, 2007.

[289] D. Farina and F. Negro, "Accessing the neural drive to muscle and translation to neurorehabilitation technologies," *IEEE Reviews in Biomedical Engineering*, vol. 5, pp. 3–14, 2012. [Online]. Available: https://doi.org/10.1109/rbme.2012.2183586

[290] A. K. Clarke, S. F. Atashzar, A. D. Vecchio, D. Barsakcioglu, S. Muceli, P. Bentley, F. Urh, A. Holobar, and D. Farina, "Deep learning for robust decomposition of high-density surface EMG signals," *IEEE Transactions on Biomedical Engineering*, vol. 68, no. 2, pp. 526–534, Feb. 2021. [Online]. Available: https://doi.org/10.1109/tbme.2020.3006508

[291] D. Farina, I. Vujaklija, M. Sartori, T. Kapelner, F. Negro, N. Jiang, K. Bergmeister, A. Andalib, J. Principe, and O. C. Aszmann, "Man/machine interface based on the discharge timings of spinal motor neurons after targeted muscle reinnervation," *Nature biomedical engineering*, vol. 1, no. 2, p. 0025, 2017.

[292] A. Gijsberts, M. Atzori, C. Castellini, H. Muller, and B. Caputo, "Movement error rate for evaluation of machine learning methods for sEMG-based hand movement classification," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, no. 4, pp. 735–744, Jul. 2014. [Online]. Available: https://doi.org/10.1109/tnsre.2014.2303394

[293] G. Li, A. E. Schultz, and T. A. Kuiken, "Quantifying pattern recognition—based myoelectric control of multifunctional transradial prostheses," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 18, no. 2, pp. 185–192, 2010.

[294] M. Ortiz-Catalan, R. Brånemark, and B. Håkansson, "Biopatrec: A modular research platform for the control of artificial limbs based on pattern recognition algorithms," *Source code for biology and medicine*, vol. 8, no. 1, pp. 1–18, 2013.

[295] Y. Geng, O. W. Samuel, Y. Wei, and G. Li, "Improving the robustness of real-time myoelectric pattern recognition against arm position changes in transradial amputees," *BioMed research international*, vol. 2017, 2017.

[296] B. Lock, K. Englehart, and B. Hudgins, "Real-time myoelectric control in a virtual environment to relate usability vs. accuracy." Myoelectric Symposium, 2005.

[297] P. M. Fitts, "The information capacity of the human motor system in controlling the amplitude of movement." *Journal of experimental psychology*, vol. 47, no. 6, p. 381, 1954.

[298] M. R. Williams and R. F. Kirsch, "Evaluation of head orientation and neck muscle emg signals as command inputs to a human–computer interface for individuals with high tetraplegia," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 16, no. 5, pp. 485–496, 2008.

[299] I. Vujaklija, V. Shalchyan, E. N. Kamavuako, N. Jiang, H. R. Marateb, and D. Farina, "Online mapping of EMG signals into kinematics by autoencoding," *Journal of NeuroEngineering and Rehabilitation*, vol. 15, no. 1, Mar. 2018. [Online]. Available: https://doi.org/10.1186/s12984-018-0363-1

[300] T. A. Kuiken, G. Li, B. A. Lock, R. D. Lipschutz, L. A. Miller, K. A. Stubblefield, and K. B. Englehart, "Targeted muscle reinnervation for real-time myoelectric control of multifunction artificial arms," *Jama*, vol. 301, no. 6, pp. 619–628, 2009.

[301] A. M. Simon, L. J. Hargrove, B. A. Lock, and T. A. Kuiken, "Target achievement control test: Evaluating real-time myoelectric pattern-recognition control of multifunctional upper-limb prostheses," *The Journal of Rehabilitation Research and Development*, vol. 48, no. 6, p. 619, 2011. [Online]. Available: https://doi.org/10.1682/jrrd.2010.08.0149

[302] M. A. Islam, K. Sundaraj, R. B. Ahmad, and N. U. Ahamed, "Mechanomyogram for muscle function assessment: a review," *PloS one*, vol. 8, no. 3, p. e58902, 2013.

[303] A. Islam, K. Sundaraj, B. Ahmad, N. U. Ahamed, and A. Ali, "Mechanomyography sensors for muscle assessment: a brief review," *Journal of Physical Therapy Science*, vol. 24, no. 12, pp. 1359–1365, 2012.

[304] S. Sikdar, H. Rangwala, E. B. Eastlake, I. A. Hunt, A. J. Nelson, J. Devanathan, A. Shin, and J. J. Pancrazio, "Novel method for predicting dexterous individual finger movements by imaging muscle activity using a wearable ultrasonic system,"

IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 22, no. 1, pp. 69–76, 2013.

[305] M. Hoshiyama, R. Kakigi, and O. Nagata, "Peripheral nerve conduction recorded by a micro gradiometer system (micro-squid) in humans," *Neuroscience letters*, vol. 272, no. 3, pp. 199–202, 1999.

[306] G. Lundborg, "TOMORROW's ARTIFICIAL HAND," *Scandinavian Journal of Plastic and Reconstructive Surgery and Hand Surgery*, vol. 34, no. 2, pp. 97–100, Jan. 2000. [Online]. Available: https://doi.org/10.1080/02844310050159927

[307] F. A. Karakostis, D. Haeufle, I. Anastopoulou, K. Moraitis, G. Hotz, V. Tourloukis, and K. Harvati, "Biomechanics of the human thumb and the evolution of dexterity," *Current Biology*, vol. 31, no. 6, pp. 1317–1325, 2021.

[308] J. A. Salomon, J. A. Haagsma, A. Davis, C. M. de Noordhout, S. Polinder, A. H. Havelaar, A. Cassini, B. Devleesschauwer, M. Kretzschmar, N. Speybroeck, C. J. L. Murray, and T. Vos, "Disability weights for the global burden of disease 2013 study," *The Lancet Global Health*, vol. 3, no. 11, pp. e712–e723, Nov. 2015. [Online]. Available: https://doi.org/10.1016/s2214-109x(15)00069-8

[309] K. Ziegler-Graham, E. J. MacKenzie, P. L. Ephraim, T. G. Travison, and R. Brookmeyer, "Estimating the prevalence of limb loss in the united states: 2005 to 2050," *Archives of Physical Medicine and Rehabilitation*, vol. 89, no. 3, pp. 422–429, Mar. 2008. [Online]. Available: https://doi.org/10.1016/j.apmr.2007.11.005

[310] T. R. DILLINGHAM, L. E. PEZZIN, and E. J. MACKENZIE, "Limb amputation and limb deficiency: Epidemiology and recent trends in the united states," *Southern Medical Journal*, vol. 95, no. 8, pp. 875–883, Aug. 2002. [Online]. Available: https://doi.org/10.1097/00007611-200208000-00018

[311] S. J. Sebastin and K. C. Chung, "A systematic review of the outcomes of replantation of distal digital amputation," *Plastic and Reconstructive Surgery*, vol. 128, no. 3, pp. 723–737, Sep. 2011. [Online]. Available: https://doi.org/10.1097/prs.0b013e318221dc83

[312] F. SCHUIND, D. ABRAMOWICZ, and S. SCHNEEBERGER, "Hand transplantation: The state-of-the-art," *Journal of Hand Surgery (European Volume)*, vol. 32, no. 1, pp. 2–17, Feb. 2007. [Online]. Available: https://doi.org/10.1016/j.jhsb.2006.09.008

[313] E. Biddiss, D. Beaton, and T. Chau, "Consumer design priorities for upper limb prosthetics," *Disability and rehabilitation: Assistive technology*, vol. 2, no. 6, pp. 346–357, 2007.

[314] H. F. Hildebrand, "Biomaterials–a history of 7000 years," *BioNanoMaterials*, vol. 14, no. 3-4, pp. 119–133, 2013.

[315] E. Biddiss and T. Chau, "Upper-limb prosthetics: critical factors in device abandonment," *American journal of physical medicine & rehabilitation*, vol. 86, no. 12, pp. 977–987, 2007.

[316] F. Cordella, A. L. Ciancio, R. Sacchetti, A. Davalli, A. G. Cutti, E. Guglielmelli, and L. Zollo, "Literature review on needs of upper limb prosthesis users," *Frontiers in neuroscience*, vol. 10, p. 209, 2016.

[317] B. Maat, G. Smit, D. Plettenburg, and P. Breedveld, "Passive prosthetic hands and tools," *Prosthetics & Orthotics International*, vol. 42, no. 1, pp. 66–74, Feb. 2018. [Online]. Available: https://doi.org/10.1177/0309364617691622

[318] S. L. Carey, D. J. Lura, and M. J. Highsmith, "Differences in myoelectric and body-powered upper-limb prostheses: Systematic literature review." *Journal of Rehabilitation Research & Development*, vol. 52, no. 3, 2015.

[319] M. A. Gonzalez, C. Lee, J. Kang, R. B. Gillespie, and D. H. Gates, "Getting a grip on the impact of incidental feedback from body-powered and myoelectric prostheses," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 29, pp. 1905–1912, 2021.

[320] P. Geethanjali, "Myoelectric control of prosthetic hands: state-of-the-art review," *Medical Devices: Evidence and Research*, pp. 247–255, 2016.

[321] J. T. Belter and A. M. Dollar, "Performance characteristics of anthropomorphic prosthetic hands," in *2011 IEEE International Conference on Rehabilitation Robotics*. IEEE, 2011, pp. 1–7.

[322] S. R. Kashef, S. Amini, and A. Akbarzadeh, "Robotic hand: A review on linkage-driven finger mechanisms of prosthetic hands and evaluation of the performance criteria," *Mechanism and Machine Theory*, vol. 145, p. 103677, Mar. 2020. [Online]. Available: https://doi.org/10.1016/j.mechmachtheory.2019.103677

[323] D. Farina and O. Aszmann, "Bionic limbs: clinical reality and academic promises," *Science translational medicine*, vol. 6, no. 257, pp. 257ps12–257ps12, 2014.

[324] A. Calado, F. Soares, and D. Matos, "A review on commercially available anthropomorphic myoelectric prosthetic hands, pattern-recognition-based microcontrollers and sEMG sensors used for prosthetic control," in *2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE, Apr. 2019. [Online]. Available: https://doi.org/10.1109/icarsc.2019.8733629

[325] Ottobock. Myo plus. Accessed 2023-05-04. [Online]. Available: https://coaptengineering.com

[326] I. B. technologies. Sense. Accessed 2023-05-04. [Online]. Available: https://www.i-biomed.com/sense.html

[327] Coapt engineering. Accessed 2023-05-04. [Online]. Available: https://coaptengineering.com

[328] U. Wijk and I. Carlsson, "Forearm amputees' views of prosthesis use and sensory feedback," *Journal of Hand Therapy*, vol. 28, no. 3, pp. 269–278, 2015.

[329] D. S. Childress, "Closed-loop control in prosthetic systems: Historical perspective," *Annals of Biomedical Engineering*, vol. 8, no. 4-6, pp. 293–303, Jul. 1980. [Online]. Available: https://doi.org/10.1007/bf02363433

[330] Z. Kappassov, J.-A. Corrales, and V. Perdereau, "Tactile sensing in dexterous robot hands," *Robotics and Autonomous Systems*, vol. 74, pp. 195–220, 2015.

[331] P. H. Chappell, "Making sense of artificial hands," *Journal of Medical Engineering & Technology*, vol. 35, no. 1, pp. 1–18, Nov. 2010. [Online]. Available: https://doi.org/10.3109/03091902.2010.526983

[332] R. B. Hellman, E. Chang, J. Tanner, S. I. Helms Tillery, and V. J. Santos, "A robot hand testbed designed for enhancing embodiment and functional neurorehabilitation of body schema in subjects with upper limb impairment or loss," *Frontiers in human neuroscience*, vol. 9, p. 26, 2015.

[333] K. Kim, J. E. Colgate, J. J. Santos-Munné, A. Makhlin, and M. A. Peshkin, "On the design of miniature haptic devices for upper extremity prosthetics," *IEEE/ASME Transactions On Mechatronics*, vol. 15, no. 1, pp. 27–39, 2009.

[334] K. A. Kaczmarek, J. G. Webster, P. Bach-y Rita, and W. J. Tompkins, "Electrotactile and vibrotactile displays for sensory substitution systems," *IEEE transactions on biomedical engineering*, vol. 38, no. 1, pp. 1–16, 1991.

[335] J. Gonzalez, H. Soma, M. Sekine, and W. Yu, "Psycho-physiological assessment of a prosthetic hand sensory feedback system based on an auditory display: a preliminary study," *Journal of neuroengineering and rehabilitation*, vol. 9, no. 1, pp. 1–14, 2012.

[336] M. Markovic, M. A. Schweisfurth, L. F. Engels, D. Farina, and S. Dosen, "Myocontrol is closed-loop control: incidental feedback is sufficient for scaling the prosthesis force in routine grasping," *Journal of NeuroEngineering and Rehabilitation*, vol. 15, no. 1, Sep. 2018. [Online]. Available: https://doi.org/10.1186/s12984-018-0422-7

[337] P.-I. Brånemark, "Osseointegration and its experimental background," *The Journal of Prosthetic Dentistry*, vol. 50, no. 3, pp. 399–410, Sep. 1983. [Online]. Available: https://doi.org/10.1016/s0022-3913(83)80101-2

[338] R. Brånemark, P. Brånemark, B. Rydevik, and R. R. Myers, "Osseointegration in skeletal reconstruction and rehabilitation," *J Rehabil Res Dev*, vol. 38, no. 2, pp. 1–4, 2001.

[339] C. J. Goodacre, J. Y. Kan, and K. Rungcharassaeng, "Clinical complications of osseointegrated implants," *The Journal of Prosthetic Dentistry*, vol. 81, no. 5, pp. 537–552, May 1999. [Online]. Available: https://doi.org/10.1016/s0022-3913(99)70208-8

[340] T. A. Kuiken, L. A. Miller, R. D. Lipschutz, B. A. Lock, K. Stubblefield, P. D. Marasco, P. Zhou, and G. A. Dumanian, "Targeted reinnervation for enhanced prosthetic arm function in a woman with a proximal amputation: a case study," *The Lancet*, vol. 369, no. 9559, pp. 371–380, 2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140673607601937

[341] H. Huang, P. Zhou, G. Li, and T. A. Kuiken, "An analysis of emg electrode configuration for targeted muscle reinnervation based neural machine interface," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 16, no. 1, pp. 37–45, 2008.

[342] J. Hoffer and G. Loeb, "Implantable electrical and mechanical interfaces with nerve and muscle," *Annals of biomedical engineering*, vol. 8, pp. 351–360, 1980.

[343] C. H. Wu, "Electric fish and the discovery of animal electricity: the mystery of the electric fish motivated research into electricity and was instrumental in the emergence of electrophysiology," *American Scientist*, vol. 72, no. 6, pp. 598–607, 1984.

[344] H. H. Zakon, D. J. Zwickl, Y. Lu, and D. M. Hillis, "Molecular evolution of communication signals in electric fish," *Journal of Experimental Biology*, vol. 211, no. 11, pp. 1814–1818, 2008.

[345] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," 2020.

[346] C. E. M. Hansel, *ESP and Parapsychology: A Critical Reevaluation*. Buffalo, New York: Prometheus Books, 1980.

[347] M. Del Giudice, "Invisible designers: Brain evolution through the lens of parasite manipulation," *The Quarterly Review of Biology*, vol. 94, pp. 249–282, 09 2013.

[348] E. Kotei and R. Thirunavukarasu, "A systematic review of transformer-based pre-trained language models through self-supervised learning," *Information*, vol. 14, no. 3, 2023. [Online]. Available: https://www.mdpi.com/2078-2489/14/3/187

[349] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, "How does batch normalization help optimization?" 2018. [Online]. Available: https://arxiv.org/abs/1805.11604

[350] G. Yang, J. Pennington, V. Rao, J. Sohl-Dickstein, and S. S. Schoenholz, "A mean field theory of batch normalization," 2019. [Online]. Available: https://arxiv.org/abs/1902.08129

[351] J. Kohler, H. Daneshmand, A. Lucchi, M. Zhou, K. Neymeyr, and T. Hofmann, "Exponential convergence rates for batch normalization: The power of length-direction decoupling in non-convex optimization," 2018. [Online]. Available: https://arxiv.org/abs/1805.10694

[352] D. Hupkes, V. Dankers, M. Mul, and E. Bruni, "Compositionality decomposed: how do neural networks generalise?" 2019. [Online]. Available: https://arxiv.org/abs/1908.08351

Paper I

Extraction of multi-labelled movement information from the raw HD-sEMG image with time-domain depth

A. E. Olsson, P. Sager, E. Andersson, A. Björkman, N. Malešević, C. Antfolk

Published in: Scientific Reports 9, 7244 (2019)

SCIENTIFIC REPORTS

OPEN

Extraction of Multi-Labelled Movement Information from the Raw HD-sEMG Image with Time-Domain Depth

Received: 10 October 2018
Accepted: 27 April 2019
Published online: 10 May 2019

Alexander E. Olsson[1], Paulina Sager[1], Elin Andersson[1], Anders Björkman[2], Nebojša Malešević[1] & Christian Antfolk[1]

In contemporary muscle-computer interfaces for upper limb prosthetics there is often a trade-off between control robustness and range of executable movements. As a very low movement error rate is necessary in practical applications, this often results in a quite severe limitation of controllability; a problem growing ever more salient as the mechanical sophistication of multifunctional myoelectric prostheses continues to improve. A possible remedy for this could come from the use of multi-label machine learning methods, where complex movements can be expressed as the superposition of several simpler movements. Here, we investigate this claim by applying a multi-labeled classification scheme in the form of a deep convolutional neural network (CNN) to high density surface electromyography (HD-sEMG) recordings. We use 16 independent labels to model the movements of the hand and forearm state, representing its major degrees of freedom. By training the neural network on 16 × 8 sEMG image sequences 24 samples long with a sampling rate of 2048 Hz to detect these labels, we achieved a mean exact match rate of 78.7% and a mean Hamming loss of 2.9% across 14 healthy test subjects. With this, we demonstrate the feasibility of highly versatile and responsive sEMG control interfaces without loss of accuracy.

The electromyogram (EMG)[1] is a time signal which describes the bioelectrical activity in skeletal muscles. The morphology of the EMG is associated with the activation, or firing, of motor units during muscle contraction. The signal is acquired by measuring the difference in electrical potential between points in (intramuscular EMG; iEMG) or on the skin covering (surface EMG; sEMG) a muscle or muscle group of interest. A high-density surface EMG (HD-sEMG)[2] is a form of sEMG where the measurement is typically acquired via a two-dimensional grid of electrodes placed on the skin of the subject. Because sEMG is a non-invasive technique it has since long been successfully applied in clinical routine, most notably for diagnosis of neuromuscular disease[3]. Since EMG is a predictor of muscle forces[4], an alternative use of sEMG is as a control signal for a system which transforms the myoelectric signal into an executable command for a device, such as a prosthesis[5], an exoskeleton[6] or a video game[7]. A system controlled by EMG signals is commonly referred to as a muscle-computer interface (MCI).

For applications of this kind, where the MCI output is to be interpreted as a movement command, it is desirable to have a *natural control scheme*, which means that the sEMG generated by one movement corresponds to MCI output encoding that very movement. To create a device with this property would require a sufficiently accurate computational estimate of the actual mapping between the space of possible sEMG signals and some space of possible movement commands. Such a mapping has proven itself elusive and difficult to model for a variety of reasons. The most important reason is that the neuromuscular processes that occur during muscle activity are inherently very complex and are at best ambiguously described by a sEMG measurement. This complexity is due to the physiological fact that each muscle is composed of a set of motor units, each which in turn is composed of many individual muscle fibers. When activated, each motor unit emits an action potential representing a sum of the electrical fields emanating from all its individual fibers[8]. As such, the signal from each sEMG electrode represents an aggregate of action potentials from adjacent motor units, additionally obfuscated by propagation

[1]Department of Biomedical Engineering, Faculty of Engineering, Lund University, Lund, Sweden. [2]Department of Hand Surgery, Skåne University Hospital, Malmö, Sweden. Correspondence and requests for materials should be addressed to C.A. (email: christian.antfolk@bme.lth.se)

through muscle-, fat- and skin tissue. Furthermore, the sEMG is notably affected by complicated types of noise, such as motion artefacts and other measurement problems[9].

Because of the complications described above, most commercially available MCI-reliant products (of which myoelectric prostheses are perhaps the most common) rely on simpler control schemes such as two-electrode proportional control[10]. The raw sEMG has traditionally been assumed to have small discriminatory power in movement classification due to its observed stochasticity, nonlinearity and unpredictability[11–13]. Because of this, previous studies aimed at improving the standards of myoelectric decoding have instead often relied on manual feature engineering[14], where each sEMG channel, commonly segmented into time blocks[15], is condensed into a set of more robust and descriptive numeric values called *features*. Such features can subsequently be used to train and evaluate a classifier. However, the creation of discriminatory sEMG features is, as in most applications of inferential statistics, a labor-intensive process that requires from the designer a good understanding of the physics of the specific problem domain. More recently, Geng *et al.*[16] defined the concept of a *sEMG image* as a grayscale image with intensity values proportional to the raw HD-sEMG measured at a single time instant and achieved unprecedented performance by applying a deep learning image classification algorithm directly on such data. From such results it can be postulated that spatial patterns correlated with movement information exists in the instantaneous raw HD-sEMG and allows for exploitation by a classifier.

Independent of feature extraction, modeling the relationship between myoelectric activity and movements is often, quite naturally, framed as a multi-class statistical classification problem. In the relevant case of hand movement recognition, the class set would consist of the set of detectable movements, while the observation instances are represented either by raw sEMG or sEMG features. While certainly a useful framework, a problem inherent to multi-class classification approaches is that the performance of any multi-class classifier devised for movement recognition by necessity decreases as the number of classes increase. This is mainly due to the fact that the EMGs associated with similar (in the sense of recruiting mutual motor units) movements are highly correlated[17,18] and thus even a sophisticated classifier used in conjunction with well-crafted features eventually lack sufficient discriminatory power. In this sense, the cardinality of the detectable movement s*et a*lways represents a compromise between classification quality (i.e. control robustness) and versatility (i.e. range of detectable movements).

In this paper we propose an alternative approach, designed to mitigate the issue of interclass correlations. We model a hand gesture not as a monolithic class, but as a combination of elements from a given set of simpler 'basis' movements. In the language of multi-label machine learning[19], the necessary and sufficient condition for a given hand movement is constituted by the presences and absences of certain mutually independent *labels*. The set of detectable movements is thus the set of possible movement label combinations. The potential value provided by the proposed framework is thought to lie mainly in its potential for scalability and stability. Importantly for prosthesis control applications, the detection of each label can be viewed as a separate classification task. Thus, in contrast to the traditional multi-class (single-label) approach, the introduction of additional classes (labels) does not directly compete for performance with those already existing. In a related sense, the multi-label approach presents a new source of stability, namely that of partial errors. When a single-label classifier infers movement intent erroneously, the prediction is by definition wholly unwanted and can result in, for example, erratic prosthesis behaviour. The output of a multi-label classifier, on the other hand, might provide a largely stable experience for the user if the majority of labels are correctly predicted most of the time, even if some individual labels are sometimes mispredicted. Lastly, and perhaps most notably, a multi-label model of this kind might ideally be able to learn to infer compound movements consisting of labels combination that do not explicitly occur in its training data, thereby massively inflating the effective range of performable movements. This ability would however require the modulation of the sEMG associated with one label combination induced by other, not previously observed, label combinations to be negligible; a property not explicitly investigated in this study.

For our experiments we adopted the use of 16 labels, shown in Fig. 1, representing flexion and extension of all digits and the wrist, thumb abduction and adduction and wrist pronation and supination. These labels were selected on the assumptions that (1) they represent movements that utilize different forearm muscles or muscle compartments[20] and (2) they, when allowed to superpose, adequately capture the major degrees of freedom possessed by the human hand. By virtue of the first assumption, they should each generate HD-sEMG signals that contain discernible patterns that, when compared pairwise across labels, are distinct enough to allow for recognition of individual labels. We implemented a deep learning[21] classifier in the form of a convolutional neural network (CNN)[22,23] with the purpose of detecting these movement labels in the HD-sEMG signal. The detection of more complicated movement is in our framework equivalent to simultaneously detecting multiple movement labels separately; some examples of such 'compound' movements that incorporate multiple active labels are shown in Fig. 1. To allow for exploitation of both spatial and temporal signal patterns, the classification procedure is performed on rescaled images with time-domain depth, i.e. sequences of consecutive sEMG images; analogous to short clips of sEMG 'video'. Previous related work[24] has been successful in demonstrating the efficacy of methods where the time-varying spatial distribution of acquired HD-sEMG is used in order to efficiently generate volitional movement commands. However, the use of structured 3-dimensional input volumes composed of stacked time slices; each depicting the instantaneous muscle state and together leveraged for the purposes of decoding movement intent, has no precedent in the MCI literature. While deep neural networks, particularly those of the convolutional kind, have been successfully utilized for classification of sEMG in the past[16,25–28], to the best of our knowledge no work has been produced to date where EMG movement decoding is treated as a multi-label classification problem.

Hand prostheses that allows for multiple degrees of freedom have existed for some time (see, for example, the Bebionic hand, Michelangelo from Ottobock and the LUKE arm from Mobius bionics). However, such prostheses are typically interacted with via sequential control strategies reliant on predetermined remnant muscles contraction patterns and as such do not operate with a natural control scheme. Development of natural control

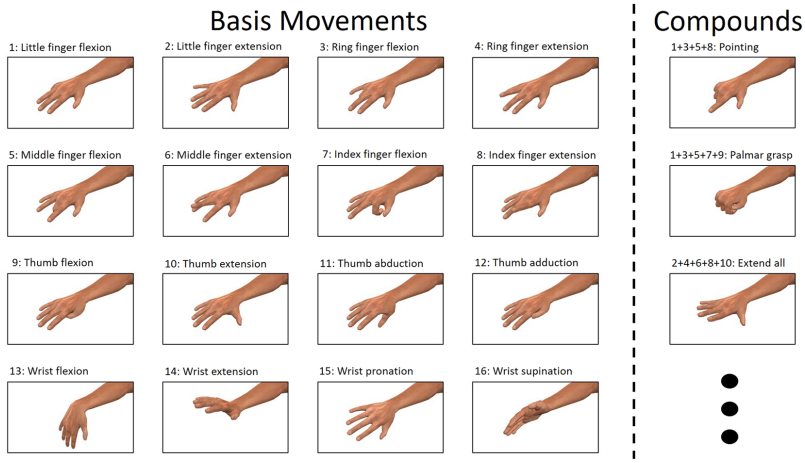


Figure 1. (left) Visualization of the label basis, constituted by 16 movements, used for multi-label classification and (right) some examples of compound movements constructed by combining labels.

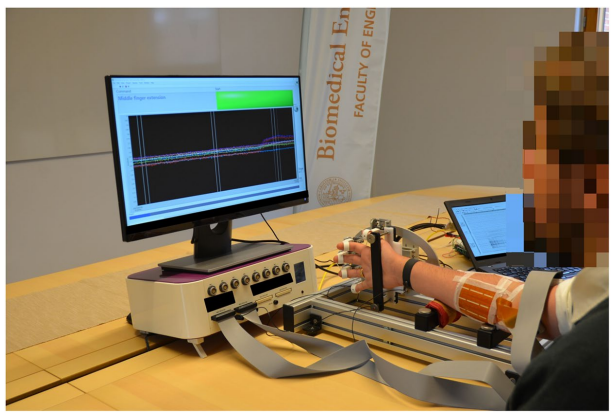


Figure 2. The experimental setup of the acquisition system.

algorithms capable of naturally utilizing the large number of available degrees of freedom is therefore positioned well to be absorbed by the growing number of mechanically very sophisticated upper limb prostheses.

Methods

Data acquisition. 14 adult and able-bodied subjects (9 male and 5 female, age range 25–57 years, median age 37 years) participated in this study. The study was approved by the Regional Ethical Review Board in Lund, Sweden and was conducted according to the tenets of the Declaration of Helsinki. All participants gave their informed written consent. During acquisition, two 8-by-8 electrode arrays with a 10 mm inter-electrode distance (ELSCH064NM3 from OT Bioelettronica, Turin, Italy) coated with conductive gel were attached to the volar part of the right forearm of the subjects. The two electrode arrays covered the skin over the extensor digitorum communis (EDC) and flexor digitorum profundus (FDP) muscles, respectively. The right hand of the subject was subsequently placed inside a custom-built rig where the forearm was comfortably resting, and the hand fixed to the rig while allowing for isometric contractions of the muscles corresponding to the set of movement labels defined for the recording protocol. While seated comfortably in a chair the subject was instructed, through a graphical user interface on a computer screen, to perform a sequence of hand movements as is shown in Fig. 2.

Each individual movement lasted for 5 seconds and was repeated 5 times, with 5 seconds of rest in between each movement repetition. The onset of each new movement was accompanied by a sound cue.

Because the number of possible movement label combinations was far too large ($2^{16} = 65536$) to be exhaustively explored in a reasonable timeframe, a subset of movement combinations was selected on the criterion that the movements should be representative of the most commonly performed hand gestures in a realistic environment[29,30]. Beyond the single label movements, every 2-label combination was recorded, excluding movements containing finger extension together with finger flexion, or with two digits separated by one or more intermediate digits. In addition, some commonly used 3, 4 and 5 label combinations were manually selected to be included: Extension of all finger, flexion of all fingers (Palmar grasp), flexion of digits 2 to 5 (i.e. excluding the thumb), Palmar grasp + pronation of wrist, extension of index finger + flexion of digits 3–5 (pointing), flexion of thumb + flexion of index finger + flexion of middle finger (3-digit pinch), 3-digit pinch + pronation of wrist, flexion of thumb + flexion of index finger (key grasp) and key grasp + pronation of wrist. This constituted 65 distinct movements in total, representing a total recording session time of approximately 1 h for each test subject.

The sEMG was amplified and sampled with an OT Bioelettronica Quattrocento (OT Bioelettronica, Turin, Italy) with a bipolar measurement scheme and a sampling frequency of $F_s = 2048$ Hz. Prior to sampling, a 10–900 Hz analogue bandpass filter was applied to each channel. A LabVIEW (National Instruments, Austin, TX) application was implemented to synchronously acquire the sEMG and the concurrent movement stimulus label set at each time sample.

Preprocessing. Once acquired, the HD-sEMG time series were digitally filtered channel-wise, first with a 2:nd order Butterworth band stop filter (48–52 Hz) for removal of power line interference and thereafter with a 20:th order Butterworth band pass filter (20–380 Hz) for suppression of noise. Samples coinciding with moments of rest were discarded at this point, justified by the assumption that rest state detection is computationally simple[31] and thus might undeservedly improve the performance of our classification scheme. In addition, samples corresponding to the first and last second of each 5 s movement repetition were discarded to eliminate effects from transient signal behavior.

The 128 filtered signals were for each sampled time point restructured into matrices of shape 16×8 with element positions corresponding to relative electrode placement. These matrices, each representing a single time instant, were individually linearly rescaled into the range [0, 1], where 0 and 1 represent the smallest and largest measured voltage, respectively, across the electrode array at the time of sampling. The resulting normalized matrices could subsequently be interpreted as digital grayscale images. With this method, a specific pixel (i.e. matrix element) value in the resulting images does not necessarily correspond to the same measured voltage across all images. The purpose of this preprocessing procedure was to extract only the spatially structured pattern of motor unit action potentials across the muscles of interest; information which was conjectured to be of greater discriminatory utility than raw myoelectric voltages. Compared to channel-wise or otherwise inter-sample normalization methods, the per-image approach taken here has the additional benefit of temporally isolating the detrimental impact of high absolute value outlier samples (e.g. noise spikes). The images were grouped into sequences of consecutive sampled time points via a sliding window of 24 samples in length and with 12 samples (50%) overlap. Thus, each sequence represents a time length of $23/F_s \approx 11$ ms. The window size and overlap preprocessing hyperparameters were selected ad-hoc, justified by the fact that they represent a minimal decision delay, operate at the (presumably) relevant timescale of sEMG fluctuations and generate a sufficiently large set of image sequence instances (approximately 110000 instances per test subject) for later training and testing of the classifier. In the last preprocessing step, each image sequence was assigned a ground truth label set of 16 bits, where each bit encodes the presence or absence of a certain label during that time interval, determined by a label-wise majority vote over the 24 sampled label sets of the sequence. Majority voting is not strictly necessary within the presented experimental framework, as we have discarded all transient signal sections with ambiguous movement affiliation, but rather serves as a universally applicable method for compressing the sequence of label sets of the window into a single label set.

The resulting image sequence instances were distributed into a training, testing and validation set as follows: instances from the 2:nd and 3:rd repetitions of each movements were used for training, instances from the 4:th repetition for testing and instances from the 5:th and final repetition for validation. Image sequences originating from the 1:st repetition of each movement were discarded since they for some subjects were wrongly labeled because the subjects occasionally forgot to perform the new movement and instead continued with the preceding movement. In all cases when a mistake occurred, an experiment supervisor successfully spotted the error and notified the subject before the 2:nd repetition began, thus preserving the integrity of repetition 2–5 for all movements and subjects. With the outlined procedure, the training, testing, and validation sets are all balanced w.r.t. the number of unique movement combinations (but not necessarily the number of individual basis movement labels). The preprocessing described here was performed via the use of custom MATLAB (The MathWorks Inc., Natick, MA) scripts.

CNN model. The structure of the CNN used in the current study was inspired by the one used by Du *et al.*[25] and is illustrated in Fig. 3. The topology and hyperparameters of the network described below were found empirically via evaluation on previously collected data and were not subject to change at any point during the current study. The input layer is a tensor of size $16 \times 8 \times 24$, representing one HD-sEMG image sequence generated in accordance with the preprocessing steps described above. It is followed by 4 convolutional blocks, connected in a feed-forward configuration, with 128, 64, 64 and 64 filters with kernel sizes 3×3, 3×3, 1×1 and 1×1, respectively. Each convolutional block follows a Convolution-BatchNorm[32]-rectified linear unit (ReLU)[23] structure, and the 3:rd and 4:th convolutional blocks have residual blocks[33] incorporated to facilitate convergence. The output from the last convolutional block is fed through a cascade of 3 fully connected blocks with a dropout[34]-fully

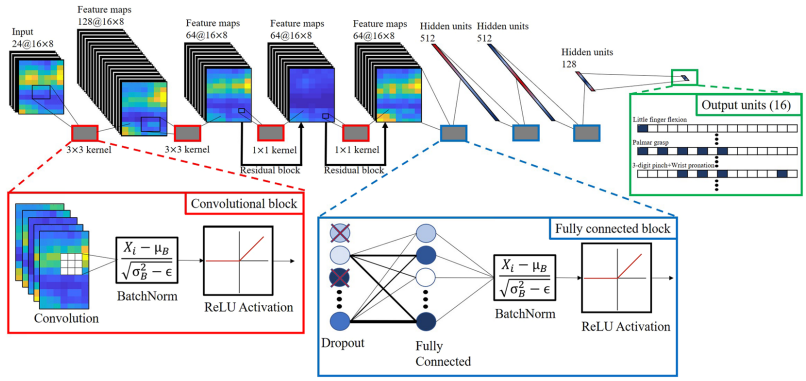


Figure 3. Illustration of the topology of the deep convolutional neural network.

connected (FC)-BatchNorm-ReLU structure with 512, 512 and 128 output neurons, respectively. All dropout layers have the dropout probability hyperparameter set to 0.5 during training. The last FC layer contains 16 output neurons, one per label, and a final sigmoid activation layer for generating label probabilities. To generate categorical label predictions, the 16 outputs of the final layer is simply compared to a probability threshold value $t \in [0, 1]$. If the i:th output element is greater than t, then the i:th label is predicted as present, otherwise absent. A higher threshold intuitively represents a higher requirement of prediction certainty on the part of the network to include a label in a prediction. The selection of t presents a directly impactful avenue of model tuning that is unavailable to (single-label) multi-class methods. In general, the threshold could be set separately for each label to achieve an arbitrarily low false positive- *or* false negative rate, but likely at the cost of a corresponding increase of the other. As such, the selection of threshold should reflect the evaluated importance of type I errors (i.e. ground truth absent labels predicted as present) relative to the importance type II errors (i.e. ground truth present labels predicted as absent). For example, in prosthesis control applications an argument could be made for a greater imperative to minimize the former (false positives), as such errors are more likely than the latter (false negatives) to be perceived by a user as directly antithetical to control stability. In our experiments we sidestep these considerations for the sake of brevity by adopting the same threshold across all labels; namely that which generates the highest exact match rate on the validation set (as is determined during model fitting). The neural model was implemented in Python 3.5 with the use of TensorFlow[35], an open-source machine learning library capable of running on graphics hardware. The model contains 4636560 learnable parameters.

Model fitting. For backpropagation[21,36], two different loss functions were evaluated: binary (per-label) cross-entropy loss[37] and BP-MLL loss; a loss function developed by Zhang *et al.*[38] specifically for training neural networks with multi-labelled output. Both loss functions were used in conjunction with a weight decay of $2{\cdot}10^{-6}$. During training, the Adam algorithm[39] with a learning rate of 0.03, mini-batch size of 3000, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\varepsilon = 10^{-8}$ was used for optimization. All learnable parameters of the network were initialized via random sampling from a truncated normal distribution with zero mean and unit variance. Training proceeded for up to 250 epochs; after every even-numbered training epoch the model was evaluated on the validation set. If no improvement over the best validation exact match rate so far was detected for 5 consecutive validations the training was halted prematurely to avoid overfitting (so-called *early stopping*). The order of the examples in the training set was randomly reshuffled prior to the onset of each new epoch. The classification probability threshold of the final network layer is determined by evaluating the fitted model on the validation set and selecting the threshold value, out of 11 candidate values sampled uniformly between 0 and 1, that maximizes exact match rate. Training was performed once per test subject training set with the fitted model applied once on the test set of the same subject to obtain performance metrics. The optimization procedure lasted for approximately 1–2 h per test subject on a desktop computer equipped with a GeForce GTX 1070 GPU (NVIDIA, Santa Clara, CA). The average time required for extraction of labels from an image sequence instance (i.e. a single network forward-pass) was 1.6 ms.

Comparison with a single-label classification scheme. In order to verify the viability of our method compared to a more conventional single-labeled classifier operating under similar conditions, an alternative network topology was evaluated on the same data set. The last two layers of this network were set as a FC layer with 65 output units (representing the set of unique recorded compound movements) followed by a softmax activation layer. Categorical prediction was performed by finding the output unit with the largest activation, as is convention in (single-label) multi-class classification, and thus no thresholding was neither required nor possible. During training, categorical cross-entropy[37] was used as the loss function to be minimized. With these final layers and loss function except, the network and the optimization procedure were identical in structure and hyperparameter selection to the model presented above. The BP-MLL loss, due to being inherently multi-label, was not utilized

	EMR	HL	JI	P	R
Cross-Entropy Loss	0.787 ± 0.064	0.029 ± 0.011	0.847 ± 0.055	0.894 ± 0.040	0.878 ± 0.054
BP-MLL Loss	0.695 ± 0.086	0.034 ± 0.013	0.827 ± 0.055	0.856 ± 0.049	0.891 ± 0.044

Table 1. Average performance metrics of the classification process across subjects. The range of each value represents its standard deviation across all subjects.

to train this network. As was the case for the multi-label network, training and evaluation was done once per test subject.

Results

A set of performance metrics relevant to multi-label classifications were selected to benchmark the predictive power of the fitted model. These were calculated once per subject on the entirety of their respective test set; the values reported in the following sections were acquired by computing the arithmetic mean and standard deviation for each metric over all subjects. Compiled results are presented in Table 1, and all results presented per subject, label and compound movement are available in Supplementary Tables 1–3. For all subjects, the optimal classification probability threshold of the neural model (derived via iterated evaluation on the validation set as described in the previous section) was determined to be 0.5 and 0.9 when trained with cross-entropy loss and BP-MLL loss, respectively. The metrics reported here were generated by models operating at these thresholds.

Exact match rate. The Exact Match Rate (EMR)[40] represents the proportion of observed image sequence instances where every single label is correctly predicted by the classifier:

$$EMR = \frac{1}{N}\sum_{t=1}^{N} 1\left(p_{t,i} = y_{t,i}\ \forall\ i\right);\quad \text{EMR} \in [0,\ 1] \tag{1}$$

where N denotes the training set cardinality, and $p_{t,i}$ and $y_{t,i}$ denotes the prediction and ground truth, respectively, of the i:th label in the t:th test instance (1 if label is present, 0 if not). $1(\cdot)$ is the indicator function, returning 1 in the case that its argument is a true condition, 0 otherwise. While closely related to the accuracy metric of conventional single-label classifiers, EMR is in a general sense much stricter the larger the set of possible labels is; a single mispredicted label in one instance marks it as failed when computing the EMR.

In the comparative case of a classifier randomly predicting each label as either present or absent with uniform probability, the expect an EMR baseline of $2^{-Q} = 2^{-16} = \frac{1}{65536} \approx 0.000015$. We achieved a mean EMR of 0.788, standard deviation $\sigma = 0.079$ when the network was trained with cross-entropy loss. With BP-MLL loss training, the resulting numbers were reduced to 0.694, standard deviation $\sigma = 0.084$. With cross-entropy loss, individual subjects reached EMR values as high as 0.879 and as low as 0.607.

Hamming loss. The Hamming Loss (HL)[40] operates on each label independently by measuring the ratio of wrongly predicted individual labels to total number of labels over all observed instances:

$$HL = \frac{1}{N}\sum_{t=1}^{N}\frac{1}{q}\sum_{i=1}^{Q} 1(p_{t,i} \neq y_{t,i}) = \frac{fp + fn}{tp + tn + fp + fn};\ \text{HL} \in [0,\ 1] \tag{2}$$

where Q = 16 is the number of possible labels. *tp*, *tn*, *fp* and *fn* denote true positive labels, true negative labels, false positive labels and false negative labels, respectively. In contrast to the other metrics presented here, a lower HL corresponds to more correctly predicted labels and is thus desirable. By its definition, HL can never exceed 1-EMR, but might be considerably smaller if the classifier often partially misclassifies instances.

In the comparative case of a classifier randomly predicting each label as either present or absent with equal probability, the expected HL would reach a baseline of 0.5. We achieved a mean HL of 0.031, standard deviation $\sigma = 0.012$ when the network was trained with cross-entropy loss. With BP-MLL loss training, the resulting numbers increased to 0.034, standard deviation $\sigma = 0.012$. With cross-entropy loss, individual subjects reached HL values as low as 0.017 and as high as 0.057.

Jaccard index. The Jaccard Index (JI)[41], sometimes referred to as *intersection-over-union*, is a statistic for measuring similarity between two sets; A and B:

$$\text{JI(A, B)} = \frac{|\text{A} \cap \text{B}|}{|\text{A} \cup \text{B}|} \tag{3}$$

here we calculate its value between the set of predictions from the test set and the test set ground truth by considering each label for each instance as a possible set element:

$$\text{JI} = \frac{\sum_{t=1}^{N}\sum_{i=1}^{Q} 1\left(y_{t,i} = 1 \wedge p_{t,i} = 1\right)}{\sum_{t=1}^{N}\sum_{i=1}^{Q} 1\left(y_{t,i} = 1 \vee p_{t,i} = 1\right)} = \frac{tp}{tp + fp + fn};\ \text{JI} \in [0,\ 1] \tag{4}$$

JI is presented as it represents a simultaneously global and fine-grained measure of classifier performance. In our experiments, we achieved a mean JI of 0.840, standard deviation $\sigma = 0.056$ when the network was trained with

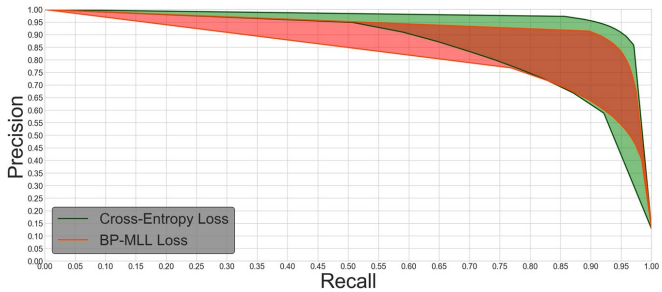


Figure 4. Precision-recall curves. The colored regions represent the different loss functions, with upper and lower bounding curves of each region corresponding to the subjects with highest and lowest EMR, respectively. The curves were plotted parametrically by linearly interpolating precision and recall calculated at 11 equidistantly spaced label detection probability thresholds between 0 and 1.

cross-entropy loss. With BP-MLL loss, the resulting numbers were reduced to 0.827, standard deviation $\sigma = 0.053$. With cross-entropy loss, individual subjects reached JI values as high as 0.908 and as low as 0.708.

Precision and recall. To investigate the possibility of classifier biases, brought about by an unbalanced training set, we calculate the Precision (P) and Recall (R) metrics[19] commonly used in binary information retrieval tasks[42,43]. Just as for the HL and JI, we extend the definition of these metrics to the multi-label problem domain by viewing each classification as $Q = 16$ independent binary instances:

$$P = \frac{\sum_{t=1}^{N}\sum_{i=1}^{Q} 1\left(y_{t,i} = 1 \wedge p_{t,i} = 1\right)}{\sum_{t=1}^{N}\sum_{i=1}^{Q} 1\left(p_{t,i} = 1\right)} = \frac{tp}{tp + fp}, \; R = \frac{\sum_{t=1}^{N}\sum_{i=1}^{Q} 1\left(y_{t,i} = 1 \wedge p_{t,i} = 1\right)}{\sum_{t=1}^{N}\sum_{i=1}^{Q} 1\left(y_{t,i} = 1\right)} = \frac{tp}{tp + fn}; \; \mathrm{P, R} \in [0, 1] \tag{5}$$

Precision represents the fraction of all predicted labels that are truly present, while recall represents the fraction of all truly present labels that is predicted as present. Over some family of classifiers with similar HL, it is therefore usual to observe an inverse relationship between P and R: A more 'lenient' classification scheme will likely retrieve more of both correct and incorrect labels, while a 'strict' scheme will retrieve fewer of both[44]. A high value of both P and R indicate that the classifier indeed exerts discriminatory power and does not simply capitalize on unbalanced data w.r.t. label abundance. The relationship between precision and recall for varying label probability thresholds in the approach taken here is presented in Fig. 4.

In our experiments, we achieved a mean precision score of 0.890, standard deviation $\sigma = 0.042$ and mean recall score of 0.868, standard deviation $\sigma = 0.055$ when the network was trained with cross-entropy loss. With BP-MLL loss, the resulting numbers were instead $P = 0.852$ ($\sigma = 0.049$) and $R = 0.891$ ($\sigma = 0.041$).

The single-labeled network. When the modified network was used to predict a single class out of the 65 recorded compound movements, only the accuracy was calculated for comparison as our selected performance metrics have no clear counterpart for single-label classification performance. Over all subject, the mean accuracy achieved was measured as 0.7813, standard deviation $\sigma = 0.068$. Individual subjects reached accuracies as high as 0.855 and as low as 0.620.

Discussion

The main aim of this study was to assess if it is possible to extract information from HD-sEMG measurements that allow for decoding of several independent hand and wrist movements simultaneously. To investigate the feasibility of such a multi-label classification approach, we implemented a deep convolutional neural network to detect up to 16 possible movement labels when given a 24 samples long sequence of sEMG images with a sampling rate of 2048 Hz. With all resulting metrics vastly outperforming a random guessing baseline, the method used in the current study can reliably be said to succeed in its task of extracting information from movement-specific spatiotemporal patterns present in the HD-sEMG. Furthermore, precision and recall scores indicate that this result is not an effect of bias induced by label imbalance or scarcity. Across all performance metrics cross-entropy loss proved slightly to moderately superior to BP-MLL loss for model convergence, despite the latter being specifically developed for use in multi-label models. This could possibly be an effect of plateaus on the high-dimensional function surface as has been previously pointed out[45] as a possible problem with BP-MLL loss.

In general, comparing the results from the current study with prior studies concerned with sEMG decoding is not straightforward, mainly since previous work has exclusively operated within a single-label framework, i.e. with a smaller set of unique classes and thus with a higher expected performance baseline. Momentarily ignoring the issues presented by comparing single-label and multi-label classifiers, a fair comparison of performance can

still only ever be made between classifiers with a similar number of inferable classes. Even so, when trained on data from our experiments and tested on the same subject, EMR values close to, and sometimes higher than, accuracies of single-label approaches were produced[26,46,47], despite our method being able to represent many more movements (65 unique movements as demonstrated in this study, and possibly many more of the untested compound movements) and EMR being a much stricter measure of performance than accuracy. A comparison with our own single-label classifier verifies that the multi-label approach does not carry with it any discernible reductions in performance to negatively offset the benefits argued for in this paper (scalability, stability and tunability), and additionally validates our method of utilizing raw sEMG image sequences for predictive purposes. Our results thus indicate that encoding hand movements with a multi-label framework could be a useful abstraction for modeling the complex relationship between the spatial and temporal variations of the sEMG and gestures constituted by multiple degrees of movement freedom.

Despite these encouraging results, some limitations on the part of our methodology need to be addressed before methods such as the one under investigation here can be implemented for use in practical circumstances, clinical or otherwise. Due to the scope of this study, data from each subject was collected during a single recording session. The presented results as they stand thus do not guarantee robustness against long-term signal variations, e.g. slight translation of electrode position over time. Further studies in this area would need to be designed to quantify and counteract such effects, or otherwise be able to tolerate small differences between the distributions generating the training-time data and the inference-time data.

The total recording session time of 1 h followed by the 1–2 h of network training necessary to fit the model might be seen as prohibitively long for the method to be of realistic utility. However, only 2 out of the 5 recorded movement repetitions were used for the subsequent fitting of the model. Hence, even within the current framework, a much more manageable recording time; less than half of that in our experiments, would suffice in order to create classifiers on par with those presented in this paper. Notably, the severity of these time constraints depends heavily on the issue of stability over time discussed in the previous paragraph, as the rate of classifier performance deterioration will determine the required frequency of recalibration. If recalibration is required often, the time required for each recalibration session must be short if the method is to have any practical feasibility. Conversely, if recalibration is only rarely required, the recording training procedure could be allowed to last for much longer. This puts further emphasis on the importance of stability over time as an object of inquiry in future studies.

One additional area of concern is that of the requirement of our method on inference time memory and computational complexity. While the extraction of the label set from an image sequence is performed in a shorter time ($<$2 ms) than the time between consecutively acquired image sequences (~6 ms), and much shorter still than would be required in a real application (~100 ms)[15], our setup has access to computational resources considerably larger than what can always be expected to be available. Future studies should focus on finding more computationally efficient multi-label classifier architectures to allow for utilization in important but resource-limited applications such as myoelectric prostheses and other types of wearable technology.

In the approach taken here, the selection of labels was based on the empirical assumption that individual finger and wrist movements should be statistically separable while also providing a good approximation of the movement of the hand and wrist state. In the future, it could be of interest to investigate more systematic approaches for the delimitation of the degrees of freedom of the hand, perhaps via the application of unsupervised machine learning, e.g. sparse autoencoders or self-organizing maps. It is likely that better basis movement labels, in the sense of generating more distinct and separable sEMG patterns while their combination space still adequately span the space of performable hand and wrist movements, can be designed.

One intriguing possibility of multi-label classifiers that was not explicitly investigated in this study is that of generalizability to unobserved label combination. Further studies should determine how well a classifier could be made to generalize learned patterns for this task. To be successful with such an approach would however require some way of ensuring that the (likely highly nonlinear) modulation of the sEMG caused by the introduction of unseen label combinations is sufficiently small to not disruptively violate the learned classification boundaries.

In closing, the topology, hyperparameters and optimization procedure of the network itself, while clearly functionally sufficient for achieving the goals stated here, can doubtlessly be improved upon in future work. As successful CNN design heuristics are lacking in the area of myoelectric pattern recognition, it could be of interest to investigate more technically sound methods of topology selection, e.g. supervised optimization procedures such as genetic algorithms.

Data Availability

Data collected during and code written for this study is available from the corresponding author on reasonable request.

References

1. Mills, K. The basics of electromyography. *Journal Of Neurology, Neurosurgery, And Psychiatry* **76**, 32–35 (2005).
2. Rojas-Martnez, M., Mañanas, M. & Alonso, J. High-density surface EMG maps from upper-arm and forearm muscles. *Journal of Neuroengineering and Rehabilitation* **9**, https://doi.org/10.1186/1743-0003-9-85 (2012).
3. Wimalaratna, H., Tooley, M., Churchill, E., Preece, A. & Morgan, H. Quantitative surface EMG in the diagnosis of neuromuscular disorders. *Electroencephalography and Clinical Neurophysiology* **42**, 167–174 (2002).
4. Hoozemans, M. & van Dieën, J. Prediction of handgrip forces using surface emg of forearm muscles. *Journal of Electromyography and kinesiology* **15**, 358–366 (2005).
5. Farina, D. *et al*. The extraction of neural information from the surface EMG for the control of upper-limb prostheses: emerging avenues and challenges. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* **22**, 797–809 (2014).
6. Jimenez-Fabian, R. & Verlinden, O. Review of control algorithms for robotic ankle systems in lower-limb orthoses, prostheses, and exoskeletons. *Medical Engineering & Physics* **34**, 397–408 (2012).
7. Kim, D. *et al*. Epidermal electronics. *Science* **333**, 838–843 (2011).

8. Farina, D., Stegeman, D. & Merletti, R. Biophysics of the Generation of EMG Signals in *Surface Electromyography: Physiology, Engineering, and Applications*, 81–105 (Wiley-IEEE Press, 2016).
9. Chowdhury, R. *et al.* Surface electromyography signal processing and classification techniques. *Sensors (basel)* **13**, 12431–12466 (2013).
10. Hubbard, S. Myoprosthetic management of the upper limb amputee in *Rehabilitation of the hand: Surgery and therapy, 4th ed.*, 1241–1252 (CRC Press, 1995).
11. Farina, D. & Merletti, R. Comparison of algorithms for estimation of EMG variables during voluntary isometric contractions. *Journal of Electromyography and Kinesiology* **10**, 337–349 (2000).
12. Norali, A., Som, M. & Kangar-arau, J. Surface electromyography signal processing and application: A review. In *Proceedings of the International Conference on Man-Machine Systems*, 11–13 (2009).
13. Goen, A. & Tiwari, D. Review of surface electromyogram signals: its analysis and applications. *International Journal of Electrical, Electronics, Communication, Energy Science and Engineering* 7, 965–973 (2013).
14. Phinyomark, A., Limsakul, C. & Phukpattaranont, P. A novel feature extraction for robust EMG pattern recognition. *Journal of Computing* **1**, 71–80 (2009).
15. Smith, L., Hargrove, L., Lock, B. & Kuiken, T. Determining the optimal window length for pattern recognition-based myoelectric control: balancing the competing effects of classification error and controller delay. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* **19**, 186–192 (2011).
16. Geng, W. *et al.* Gesture recognition by instantaneous surface EMG images. *Scientific Reports* **6**, https://doi.org/10.1038/srep36571 (2016).
17. Ju, Z., Ouyang, G. & Liu, H. EMG-EMG correlation analysis for human hand movements. In *IEEE Workshop on Robotic Intelligence in Informationally Structured Space (RiiSS)*, 38–42 (2013).
18. Semmler, J. & Nordstrom, M. A comparison of cross-correlation and surface emg techniques used to quantify motor unit synchronization in humans. *Journal of neuroscience methods* **90**, 47–55 (1999).
19. Madjarov, G., Kocev, D., Gjorgjevikj, D. & Džeroski, S. An extensive experimental comparison of methods for multi-label learning. *Pattern recognition.* **45**, 3084–3104 (2012).
20. Fleckenstein, J. L., Watumull, D., Bertocci, L. A., Parkey, R. W. & Peshock, R. M. Finger-specific flexor recruitment in humans: depiction by exercise-enhanced MRI. *Journal of Applied Physiology* **72**, 1974–1977 (1992).
21. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521**, 436–444 (2015).
22. LeCun, Y. Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems*, 396–404 (1990).
23. Krizhevsky, A., Sutskever, I. & Hinton, G. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 1097–1105 (2012).
24. Ison, M., Vujaklija, I., Whitsell, B., Farina, D. & Artemiadis, P. High-Density Electromyography and Motor Skill Learning for Robust Long-Term Control of a 7-DoF Robot Arm. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* **24**, 424–433 (2016).
25. Du, Y., Jin, W., Wei, W., Hu, Y. & Geng, W. Surface EMG-Based Inter-Session Gesture Recognition Enhanced by Deep Domain Adaptation. *Sensors (Basel)* **17**, https://doi.org/10.3390/s17030458 (2017).
26. Atzori, M., Cognolato, M. & Müller, H. Deep learning with convolutional neural networks applied to electromyography data: a resource for the classification of movements for prosthetic hands. *Frontiers in Neurorobotics* **10**, https://doi.org/10.3389/fnbot.2016.00009 (2016).
27. Ahsan, M., Ibrahimy, M. & Khalifa, O. Electromygraphy (EMG) signal based hand gesture recognition using artificial neural network (ANN). In *4th International Conference on Mechatronics (ICOM)*, 1–6 (2011).
28. W. Wei *et al.* A multi-stream convolutional neural network for sEMG-based gesture recognition in muscle-computer interface. *Pattern Recognition Letters*, https://doi.org/10.1016/j.patrec.2017.12.005 (2017).
29. Feix, T., Romero, J., Schmiedmayer, H., Dollar, A. & Kragic, D. The GRASP Taxonomy of Human Grasp Types. *IEEE Transactions on Human-Machine Systems* **46**, 66–77 (2016).
30. Cutkosky, M. On grasp choice, grasp models, and the design of hands for manufacturing tasks. *IEEE Transactions on robotics and automation* **5**, 269–279 (1989).
31. Staude, G., Flachenecker, C., Daumer, M. & Wolf, W. Onset detection in surface electromyographic signals: A systematic comparison of methods. *EURASIP Journal on Advances in Signal Processing* **2**, 67–81 (2001).
32. Ioffe, S. & Szegedy, C. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *Proceedings of the International Conference on Machine Learning* 448–456 (2015).
33. He, K., Zhang, X., Ren, S. & Sun, J. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778 (2016).
34. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* **15**, 1929–1958 (2014).
35. Abadi, M. *et al.* TensorFlow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, 265–283 (2016).
36. LeCun, Y., Bottou, L., Orr, G. & Müller, K. Efficient BackProp in *Neural Networks: Tricks of the trade*, 9–48 (Springer, 2012).
37. Goodfellow, I., Bengio, Y. & Courville, A. Machine Learning Basics in *Deep Learning* (MIT press, 2016).
38. Zhang, M. & Zhou, Z. Multilabel Neural Networks with Applications to Functional Genomics and Text Categorization. *IEEE Transactions on Knowledge and Data Engineering* **18**, 1338–1351 (2006).
39. Kingma, D. & Ba, J. Adam: a method for stochastic optimization. Preprint at, https://arxiv.org/abs/1412.6980 (2014).
40. Sokolova, M. & Lapalme, G. A systematic analysis of performance measures for classification tasks. *Information Processing and Management* **45**, 427–437 (2009).
41. Levandowsky, M. & Winter, D. Distance between sets. *Nature* **234**, 34–35 (1971).
42. Harbour, J. *et al.* Reporting methodological search filter performance comparisons: A literature review. *Health Information and Libraries Journal* **31**, 176–194 (2014).
43. Calijorne Soares, M. & Parreiras, F. A literature review on question answering techniques, paradigms and systems. *Journal of King Saud University - Computer and Information Sciences*, https://doi.org/10.1016/j.jksuci.2018.08.005 (2018).
44. Buckland, M. & Gey, F. The relationship between recall and precision. *Journal of the American Society for Information Science* **45**, 12–19 (1994).
45. Nam, J., Kim, J., Mencía, E., Gurevych, I. & Fürnkranz, J. Large-Scale Multi-label Text Classification — Revisiting Neural Networks. In *Joint European conference on machine learning and knowledge discovery in databases*, 437–452 (2014).
46. Atzori, M. *et al.* Electromyography data for non-invasive naturally-controlled robotic hand prostheses. *Scientific Data* **1**, 140053 (2014).
47. Amma, C., Krings, T., Böer, J. & Schultz, T. Advancing muscle-computer interfaces with high-density electromyography. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, 929–938 (2015).

Acknowledgements
This research is supported by the EU-funded DeTOP Poject (EIT-ICT-24-2015, GA no. 687905), the Promobilia Foundation and the Crafoord Foundation.

Author Contributions
A.O. designed and implemented the data preprocessing and machine learning algorithms, performed the analysis of results, wrote the manuscript and prepared all figures. P.S. and E.A. performed data acquisition and parts of the data preprocessing. A.B. designed the study and aided in data acquisition. N.M. wrote the software for data acquisition, designed the study and supervised data acquisition. C.A. conceived the idea, designed the study and wrote the manuscript. All authors read and approved the final manuscript.

Additional Information
Supplementary information accompanies this paper at https://doi.org/10.1038/s41598-019-43676-8.

Competing Interests: The authors declare no competing interests.

Publisher's note: Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Paper II

Automatic discovery of resource-restricted Convolutional Neural Network topologies for myoelectric pattern recognition

A. E. Olsson, A. Björkman, C. Antfolk

Published in: Computers in Biology and Medicine 120, 103723 (2020)

Computers in Biology and Medicine 120 (2020) 103723

Contents lists available at ScienceDirect

Computers in Biology and Medicine

journal homepage: http://www.elsevier.com/locate/compbiomed

Automatic discovery of resource-restricted Convolutional Neural Network topologies for myoelectric pattern recognition

Alexander E. Olsson [a,*], Anders Björkman [b,c], Christian Antfolk [a,**]

[a] Dept. of Biomedical Engineering, Faculty of Engineering, Lund University, Lund, Sweden
[b] Dept. of Hand Surgery, Lund University, Skåne University Hospital, Malmö, Sweden
[c] Wallenberg Center for Molecular Medicine, Lund University, Lund, Sweden

ARTICLE INFO

Keywords:
Electromyography
Myoelectric pattern recognition
Myoelectric control
Muscle-computer interfaces
Model selection
Convolutional neural networks
Machine learning
Deep learning

ABSTRACT

Convolutional Neural Networks (CNNs) have been subject to extensive attention in the pattern recognition literature due to unprecedented performance in tasks of information extraction from unstructured data. Whereas available methods for supervised training of a CNN with a given network topology are well-defined with rigorous theoretical justification, procedures for the initial selection of topology are currently not. Work incorporating selection of the CNN topology has instead substantially been guided by the domain-specific expertise of the creator(s), followed by iterative improvement via empirical evaluation. This limitation of methodology is restricting in the pursuit of naturally controlled muscle-computer interfaces, where CNNs have been identified as a promising research avenue but effective topology selection heuristics are lacking. With the goal of mitigating ambiguities in topology selection, this paper presents a systematic approach wherein we apply a novel evolutionary algorithm to search a space of candidate topologies. Furthermore, we constrain the search-space by excluding topologies with excessive inference-time computational complexity, making the obtained results implementable in embedded systems. In contrast to manual topology design, our algorithm requires the user to only specify a relatively small set of intuitive hyperparameters. To validate our approach, we use it in order to create topologies for myoelectric pattern recognition via movement decoding of surface electromyography signals. By collating offline classification accuracies obtained from experiments on a collection of publicly available databases, we demonstrate that our method generates computationally lightweight topologies with performance comparable to those of available alternatives.

1. Introduction

Improvements related to the efficacy of the relevant hardware and algorithms have, during the preceding decade, led to a noticeable increase in research concerning the class of machine learning methods colloquially known as Deep Learning [1,2]. Among such methods, few have contributed more to the newly emerged feasibility of methodology based on Pattern Recognition (PR) than the Convolutional Neural Network (CNN) [3]. Since 2013, every winning participant of the annual Large Scale Visual Recognition Competition [4] have opted for some form of CNN-based approach [5], lending credence to a view of the CNN model as the gold standard for tasks involving image recognition. Distinguished contributions to this competition, such as AlexNet [6], GoogleNet [7], and ResNet [8] succinctly illustrate many of the relatively recent algorithmic innovations in the field that have drastically reduced and sometimes virtually eliminated problems (vanishing gradients, covariate shift, etc.) traditionally afflicting aspiring domains of neural computation. Parallel to this development in the computer vision discipline, the incorporation of CNN models has turned out to be a successful strategy in many somewhat adjacent areas where information extraction from unstructured signals is of value, e.g. speech recognition [9], text parsing [10], and drug discovery [11]. One such area in which Deep Learning in general and CNNs in particular have rapidly advanced concerns the task of myoelectric pattern recognition [12]; a task for which progress holds great promise for the future development of muscle-computer interfaces. A muscle-computer interface is here defined as systems which, in a wide sense, extracts a *control signal* from a given set of neuromuscular signals, i.e. an electromyogram (EMG) [13].

* Corresponding author.
** Corresponding author.
E-mail addresses: alexander.olsson@bme.lth.se (A.E. Olsson), christian.antfolk@bme.lth.se (C. Antfolk).

https://doi.org/10.1016/j.compbiomed.2020.103723
Received 16 January 2020; Received in revised form 6 March 2020; Accepted 21 March 2020
Available online 25 March 2020

Salient technical domains in which functional myoelectric interfacing of this kind currently play an important role or in the future might have potential to play important roles include video game interfacing [14], exoskeleton piloting [15] and prosthesis control [16]. In the primarily envisioned use case of upper limb prosthetics, the control signal extracted by the interface is to be interpreted as a movement instruction that is subsequently fed to a powered actuator, i.e. a robotic arm. With the end user in mind, it is for this kind of application highly desirable with a *natural control scheme*; here meaning that the EMG generated by one's skeletal muscles while performing a movement corresponds to output encoding that very movement. This notion is corroborated by surveys conducted with amputees, where intuitive and precise motor control is reported as a prioritized feature in the development of prosthetic limbs [17–19].

Currently commercially available, myoelectrically controlled hand prostheses often allow for motorized actuation of multiple independent degrees of freedom (DoFs) with a finely resolved range of exertable force levels almost on par with that of the natural human hand [20,21]. However, for almost all such modern prostheses, the methods for inferring the intent of the user and subsequently controlling the aforementioned functionality are still fundamentally dissimilar to the corresponding process in the natural hand. In the typical case, a sparse setup of electrodes records the surface EMG (sEMG) produced by the contraction of an agonist-antagonist muscle pair in the residual arm of the prosthesis user. The averaged amplitude of the recorded signal is used for direct binary or proportional control of a single DoF [22], while techniques based on e.g. cocontraction (simultaneous contraction of antagonistic muscle pairs), sequential contractions, or non-EMG inputs [23] are used to cycle between different DoFs. Due to their robustness to noise and stability over time, such control schemes are unlikely to be perceived as uncompliant or to behave erratically during usage. However, they clearly do not possess the desired naturalness and are furthermore restricted to efferent control of only one DoF at a time, severely limiting real-time dexterity. To bridge the growing gap separating the mechanical sophistication of available prostheses and their comparatively primitive control methods, efforts have been made to instead bring methods based on pattern recognition to bear on the problem (see [24–26] for reviews). In the resulting body of work, the task of decoding movement intent from neuromuscular activity is naturally formulated as a task of *supervised learning*. Formulated with the terminology of inferential statistics, this framework views sEMG, or descriptive *features* [27] extracted from sEMG, as observations of independent variables, while the corresponding observations of dependent variables consist of an encoding of the wrist- and hand motions cooccurring with sEMG acquisition. The task of myoelectric pattern recognition is then reducible to deducing the values of the dependent variables given an instance of independent variables. Encoding of motion can be either continuous, e.g. quantification of individual muscle forces, or categorical, e.g. the presence or absence of mutually exclusive gestures, in which case the task becomes that of statistical *regression* or *classification*, respectively, with much of the contemporary technical literature focused on the latter alternative. In this framework, a diverse set of predictive algorithms including versions of k-nearest neighbours [28–30], linear discriminant analysis [31], support vector machines [32,33], hidden Markov models [34,35], and decision trees/random forests [36], have been considered as potential candidates for achieving reliable prosthetic control. Whereas previous studies (with few and distinct movements) have managed to generate impressive classification accuracies exceeding 95% [37], extremely few of these methods have yet been judged to exhibit the stability, validity and reliability required for widespread use in daily routine.

As in many other disciplines, the inability of 'traditional', feature-based machine learning to conclusively offer a satisfactory solution has led researchers to investigate the application of Deep Learning. From a high-level perspective, a muscle-computer interface operating with a natural control scheme must, explicitly or implicitly, contain within it a computable function mapping some subspace of physically possible EMG signals to efferent commands (corresponding to some subspace of natural movements). Given that such a mapping exists and has acceptable fidelity, i.e. sufficient information about movement intent is embedded in the EMG signal, the task can be formulated as approximating the mapping to a satisfactory degree. From this view, feed-forward multilayer artificial neural networks, in their capacity as universal function approximators [38], are intuitively a promising class of algorithms to pursue. Although research is still in its infancy, a number of studies ([39–49], among others) incorporating deep neural models have persuasively argued that reliable discernment of movement information from raw sEMG measurements is possible, with performance often surpassing that of classical methods in one or several respects. Related to their success in computer vision, CNNs have been applied to high-density sEMG (HD-sEMG) [50], where surface electrodes arranged in a 2D grid are used to capture a *sEMG image*, to exploit latent spatially encoded signal characteristics and achieve unprecedented performance [40,47,48]. Recently, recurrent models have been used to the same effect [43]. Furthermore, unsupervised Deep Learning such as sparse autoencoders [45] and deep belief networks [46] have been used to find EMG embeddings of discriminatory power comparable to handcrafted features.

The ability of all of the methods described above to operate directly on sampled myoelectric voltages and thus circumvent manual feature engineering is encouraging in the sense that no arbitrary decision regarding compression of spatiotemporal information needs to be made but is instead automatically deduced from the signals themselves. However, this welcome simplification is accompanied by a drastic increase in the number of arbitrary tuning considerations inherent to the algorithms themselves. For example, to use a CNN requires not only the specification of training-time hyperparameters (learning rate, minibatch size, etc. [1]), but also a selection of a network *topology*. Sometimes denoted *architecture* in the literature, the term topology in this context simply refers to the sequence of *layers*, each possibly having hyperparameters of its own, the interlace of which constitute the network. With some recent algorithmic innovations such as AdaNet [51] exempt, the topology is typically treated as a hyperparameter to be selected prior to implementation. In computer vision, successful topologies have typically been the result of strenuous research, often stretching over many years [52]. This presents an inconvenience to new prospective domains of CNN application (such as myoelectric pattern recognition) where no validated topology design heuristics exist *a priori*, and no guarantee can be provided that the quality of previously successful topologies will be conserved when transferred to the new problem domain. More specific to the field of prosthetics, limitations of memory and processing power in the inference-time environment of algorithms put computational restrictions on topologies, making transfer of results from adjacent fields yet more difficult.

Naturally, we are not the first to notice the intractable properties of CNN construction, and a body of work discussing methods for *automatic topology selection* has grown rapidly for some time (see, for example [53–56]). One of the earliest attempts at this task is that of neuro-evolution [57]; a class of methods for generating well-behaving network topologies and/or parameters/hyperparameters based on natural selection. For some time, the reigning perception has been that such automatically generated networks are incapable of matching the performance of their manually designed counterparts [52]. However, recent studies undertaken to investigate novel evolutionary approaches to topology search have, backed by promising results, argued against this view [52]. Such studies have as of yet mostly focused on tasks of traditional image recognition (i.e. pertaining to computer vision) and have not been concerned with finding suitable topologies for domains where reliable alternatives are lacking. Thus, issues such as domain transferability and limited resources are seldom addressed.

In an attempt to relieve the problems outlined so far, we here propose a novel procedure aimed at discovering new and promising CNN

topologies without requiring domain-dependent expertise from the designer. To this end, we introduce an evolutionary algorithm (EA) [58] to generate topologies intended for sEMG decoding in muscle-computer interfaces. Whereas genetic algorithms have been applied in the context of sEMG decoding previously [59], to the best of our knowledge no attempts of evolving neural network topologies for myoelectric pattern recognition been undertaken so far. The presented EA exhibits some deviations from previous contributions chosen in order to reflect the nature of the problem domain better. Firstly, the algorithm was constructed in a way that excludes topologies with too large hardware requirement, allowing for direct implementation of results in a resource-restricted embedded environment. In most other regards, the search-space is unrestricted as to not impose potentially false preconceived beliefs about what constitutes a well-behaving topology. Secondly, we introduce mutation operators, a mutation imposition formula, and a topology encoding scheme method which captures and explores the scope of plausible CNNs well, thereby facilitating a comparatively rapid convergence. Lastly, we introduce a novel topology regularization method, referred to here as *neural cleansing*, which periodically removes superfluous layers that do not contribute to network expressivity throughout the search. By doing so, we free up computational resources that can subsequently be used for more productive topology alterations. By evaluating the approach on publicly available databases we ensure that results can be verified and that comparison with related methods is possible. With this work, we hope to (1) demonstrate the applicability of evolutionary topology search and its usefulness for successful deployment of myoelectric pattern recognition based on Deep Learning in real environments and (2) provide an indication of the feasibility of evolutionary search for neural model selection in biosignal processing in general.

2. Methods

This section outlines the technical details of the proposed method for topology discovery.

2.1. System overview

The basic structure of the topology search procedure, as is illustrated in Fig. 1, is that of a fixed population size EA with synchronous generational progression and no crossover. With terminology from evolutionary biology, the highest-level abstraction in this operational framework is that of the *population* of candidate solutions. The population consists of a positive integer C_p (here a hyperparameter) *individuals*, each with an accompanying *genome*. The genome, in turn, uniquely encodes the specific *solution* that the individual carrier represents. In the relevant context of topology search, a solution is synonymous with a CNN topology.

The algorithm is initialized via the creation of a (homogeneous) population of individuals with genomes that represent trivial solutions which are, presumably, far from optimal. In the implementation used in this study, the genomes of the initial populations are selected to encode the arguably simplest possible classificatory feed-forward neural model, i.e. that of an input layer, a single *n*-way fully connected layer, and an output softmax layer, where *n* is the number of classes. With an initial population created in this way, the algorithm repeats, in sequence, the stages of *fitness evaluation*, *selection* and *mutation*. Pseudocode formalization of this iterative procedure is shown in Fig. 2. At the fitness evaluation stage, each individual is assigned a fitness value representing the quality of the solution it represents. At the selection stage, individuals are appointed as parents on the basis of attained fitness. At the

Require $C_p \in \mathbf{N}^+$ (population cardinality)
Require $G \in \mathbf{N}^+$ (total number of generations)
Require $T_c \in \mathbf{N}^+$ (period of neural cleansing)
Require X_t (Annotated training data)
Require X_v (Annotated validation data)
create population of individuals $P \leftarrow \{I_1, I_2, \ldots, I_{C_p}\}$
set initial fitnesses $f_i \leftarrow 1$ of $I_i \forall$ i
set current generation $g \leftarrow 1$
While $g \leq G$ **do**
    create new empty population $P_{new} \leftarrow \emptyset$
    While $\|P_{new}\| < C_p$ **do**
        $I \leftarrow$ *select*(P)
        $I_{new} \leftarrow$ *mutate*(I)
        append I_{new} to P_{new}
    end While
    $P \leftarrow P_{new}$
    If g is divisible by T_c **then**
        $P \leftarrow$ *neural-cleansing*(P)
    end If
    For each I in P **do**
        $f_i \leftarrow$ *evaluate*(I_i, X_t, X_v)
    end For
    increment generation $g \leftarrow g + 1$
end While
find $I_{best} \leftarrow I_i$, where $i = \underset{i}{\text{argmax}}(f_i)$
Return topology of I_{best}

Fig. 2. Pseudocode describing the basic structure of the topology search EA.

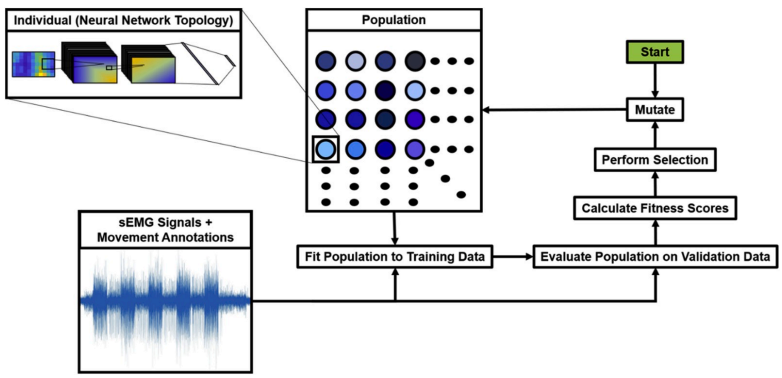


Fig. 1. Flowchart illustrating the EA approach to discovery of CNN topologies for MPR.

mutational stage, the genomes of the selected parents are altered before they are used as the basis for new individuals which in turn constitute a new population, replacing the prior. This cycle is repeated for a positive integer *G* (here a hyperparameter) *generations* to iteratively improve the quality of the genome-encoded solutions present in the population via consistent selective pressure. The details of the utilized encoding-, fitness evaluation-, mutation-, and selection schemes are explored in the following sections.

2.2. Topology encoding

The genome of an individual contains all information required in order to construct, train and test the performance of the CNN topology it represents. In the genome, topology is represented as a dynamic array of *layers*, with the internal order of the array elements representing the order in which their respective operations are performed on the input to the represented feed-forward network. Each topology list element is in turn represented by a static array of parameters, unambiguously defining the operating characteristics of the encoded layer instance. Encodable layer types, with respective parameters, are shown in Fig. 3 and given by:

- **Fully Connected** [1] with parameters:
  - output neurons (integer).
- **Convolutional** [3] with parameters:
  - number of kernels (integer),
  - kernel width (integer)
  - kernel height (integer)
  - horizontal stride (integer)
  - vertical stride (integer)
- **Pooling** [3] with parameters:
  - kind ('average' [3] or 'pooling' [6])
  - window width (integer)
  - window height (integer)
  - horizontal stride (integer)
  - vertical stride (integer)
- **Activation** with parameter:
  - kind ('ReLU' [6], 'tanh' [1], or 'softmax' [1]).
- **Dropout** [60] with hyperparameter:
  - probability (floating-point number)
- **Batch Normalization** [61] (no parameters).

In this study, convolutional layers use zero-padding to keep the spatial dimensions of their output equal to those of their input. In addition to the structural information, the genome contains the specification of *initializers*, *regularizers*, and the *optimizer* used to train the network. The reason for their inclusion in the genome is that the optimal topology (in the sense of maximizing fitness) is highly dependent on the configuration of such parameters during network training. Because we are concerned only with maximizing the performance of the topology in the absolute sense, initializers, regularizers and optimizer are allowed to evolve in tandem with the topology. The initializers are encoded as a floating-point number per fully connected and convolutional layer in the topology, representing the standard deviation of the normal distribution from which the layers initial weights are sampled. The regularizers are encoded as a floating-point number per fully connected and

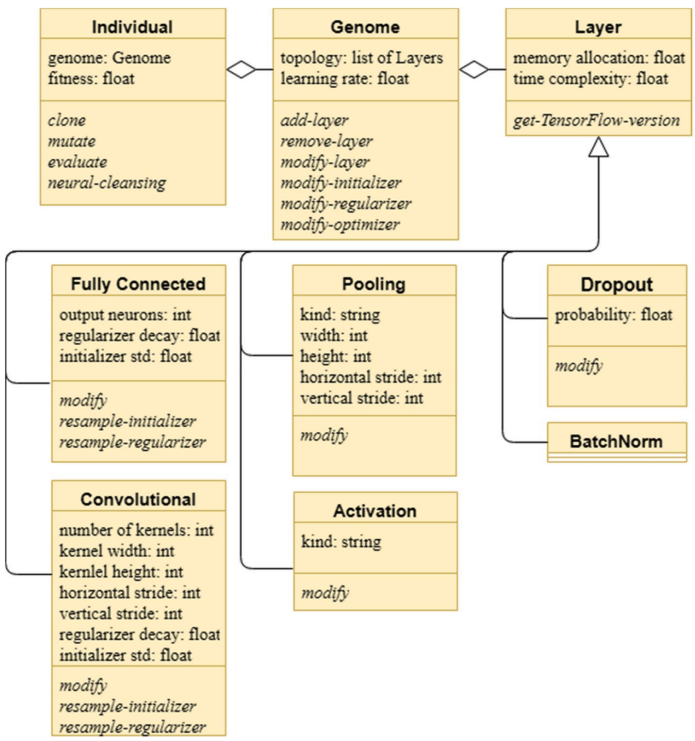


Fig. 3. Class diagram of the genetic encoding scheme.

convolutional layer in the topology, representing the rate of their respective weight decay [1]. Lastly, the genome encodes the optimizer as a floating-point number representing learning rate.

2.3. Fitness evaluation

To calculate the fitness of a given individual (topology) in the population, a training set and a validation set of sEMG signals with movement annotations, collected from one or more human test subjects, are needed. One CNN is instantiated, as specified by the genome of the individual, per test subject. Each such network is trained on motion ground-truth annotated sEMG data from a single subject each, and subsequently evaluated on validation data from the same subject. Backpropagation-based training is performed with the Adam algorithm [62] with genome-dependent learning rate, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$ and with a minibatch size of 512 for 10 training epochs, minimizing categorical cross-entropy loss [1]. The batch size and total epochs values are held constant; allowing them to evolve could lead to unbounded growth and thus long evaluation times without a corresponding increase in evaluation quality. At the onset of training, network weights are initialized independently by sampling from a normal distribution with zero mean and a layer-specific standard deviation (encoded by the genome). Biases are globally initialized to zero. L_2 regularization of weights is applied during training, with the rate of weight decay, individual to each layer, encoded by the genome. Following inference on the validation set, fitness of individual I_i is computed via (1).

$$f_i = 2^{\frac{\widehat{a}_i}{a_2}} \tag{1}$$

Here, $\widehat{a}_i$ is the arithmetic mean of the classification accuracies achieved by the topology of the individual I_i on some validation data, averaged across all test subjects. This formulation has as a consequence that additive increases in validation accuracy will always be rewarded by multiplicative increases in fitness, partially counteracting the diminishing returns inherent to marginal accuracy gains. The value of the hyperparameter $a_2 \in (0, \infty]$ is the *fitness-doubling accuracy difference* and represents the additive increase in accuracy required of an individual to double its fitness.

2.4. Selection

The proposed algorithm incorporates a type of proportionate selection with elitism [58]. As a first step in this procedure, the individual with highest fitness (as determined previously) present in the population of the current generation is without any mutations directly transferred ('cloned') into the population of the next generation. This simple step ensures that 'genetic breakthroughs' are much less prone to accidentally not get retained during generational progression, all with minimal loss to exploratory potency of the optimization procedure. Following this initial exception, individuals of the next generation are iteratively created by repeatedly selecting a random individual from the population of the current generation for breeding. Each such selection results in a single new offspring individual with a genome which is a mutated version of that of its parent. The probability p_i of selecting a given individual I_i for breeding is at each iteration directly (linearly) proportional to the fitness f_i achieved by that individual during the preceding evaluation step:

$$p_i = \frac{f_i}{\sum_j f_j} \tag{2}$$

There is no hard limit (other than the globally set population size $C_p \geq i \; \forall \; i$) to the number of times a single individual can be selected as a parent; but comparatively high performing individuals are likely to have numerous derivatives of their genome permeate the population.

2.5. Mutation

Each time an individual is selected (in the context of the breeding procedure outlined previously), a nonnegative integer m is sampled from the Poisson-distributed random variable M with probability mass function f_M given by:

$$f_M(m) = S^{\cdot}\frac{(-logS)^m}{m!} \tag{3}$$

The sampled value m represents the number of mutations to be performed on the genome of the individual in question. The mutation procedure utilizes a hyperparameter, referred to as *mutational stability*, which is denoted by $S \in (0, 1]$. S can here be interpreted as the probability of no mutation happening, which follows from simplifying (3) in the case of $m = 0$, whereby $\Pr(M = 0) = f_M(0) = S^{\cdot}\frac{(-\log(S))^0}{0!} = S$. If $m = 0$, the offspring individual will be an exact copy ('clone') of the parent. If instead $m > 0$, m mutations are selected sequentially and at random from the set of allowed mutations, each with a respective probability (tuneable as hyperparameters). If $m > 1$, the same mutation can be performed multiple times. The set of allowed mutation operators are given below. Here, *log-uniform distribution* is taken to mean the distribution of a random variable which has a uniformly distributed logarithm (See Appendix A for details regarding layer creation and modification routines.).

- *add-layer*: Select a random layer type from the set of allowed layers (as presented in section II·B) with uniform probability. Insert a layer of that type at a random (uniform) index (prior to the terminal FC and softmax layers) in the genome topology array.
- *remove-layer*: Select a random layer to delete with uniform probability over all layer present in the network. The terminal FC and softmax activation layers are necessary for meaningful network output and cannot be removed.
- *modify-layer*: Randomly select a layer in the topology with all eligible layers weighted uniformly. All layers, except the terminal FC layer, the softmax layer, and those of the BN kind, are eligible to this mutation. The selected layer has its parameters modified according to a routine specific to each layer type.
- *modify-initializer*: The standard deviation of the normal distribution used to sample the initial weights is updated for a random FC or convolutional layer (probability uniform over all candidates in the topology). The new standard deviation is sampled from the log-uniform distribution between 10^{-3} and 10^{0}.
- *modify-regularizer*: The rate of weight decay for a random FC or convolutional layer (probability uniform over all candidates in the topology) is resampled from the log-uniform distribution between 10^{-9} and 10^{0}.
- *modify-optimizer*: The learning rate to be used during the training phase of the topology is resampled from the log-uniform distribution between 10^{-6} and 10^{0}.

If the selected mutation is undefined for the current topology (e.g. *remove-layer* or *modify-layer* in cases when no layers other than the terminal fully connected and softmax layers exist in the topology), a new attempt at picking a mutation is made. The procedure is repeated until m legal mutations have been picked; ensuring that the genome has been subject to exactly m mutations before the offspring individual is generated. For each mutation, the allocation of memory required to instantiate the CNN topology of the postmutational genome and the inference-time computational complexity of ditto are estimated. If any of these two values exceeds their respective *a priori* set threshold allowance value (EA hyperparameters $C_t^{max} \in [0, \infty]$ and $C_s^{max} \in [0, \infty]$), the mutation is not performed. Just as in the case of impossible and undefined mutations, a new mutation will (repeatedly, if necessary) be resampled from the set of possible mutations until a legal mutation is selected. Consequently, at

no point during the evolutionary procedure will genomes containing topologies with too large (estimated) requirements on computational resources exist. (The approach to automatic estimation of resources required by a topology is outlined in Appendix B.)

2.6. Neural cleansing

The EA as it has been presented so far exerts no direct selective pressure for computational efficiency of candidate topologies. During the search, ineffective layers (in the sense of contributing neither positively nor negatively to fitness) can thus remain in the topology of an individual indefinitely and adversarially affect both the per individual time complexity (thus increasing search time) and the efficacy of the ultimate result. This effect is mitigated by having the probability of the *remove-layer* mutation be greater than that of the *add-layer* mutation, but only to a limited extent. Due to the hard limits on memory footprint and time complexity imposed during mutation, there is additional risk of reaching genetic 'dead-ends', where improbable sequences of mutations are necessary in order to improve fitness at all. Outright prohibition of mutations resulting in ineffective layers, as is done with e.g. topologies exceeding the restrictions on computational complexity, would rule out this possibility, but would also result in a more restricted search space as some topologies becoming impossible to reach. To reduce the risks of excessive 'mutation load' while retaining sufficient exploratory leniency, we instead introduce a novel type of topology regularization technique which we refer to as *neural cleansing*. At equigenerational intervals throughout the search, the genomes of the entire populations are subject to a routine which removes *superfluous* layers. A layer is regarded as superfluous if the network prior to and after its removal can, during training, learn weights and biases which perform the exact same input-output mapping. With this definition, all layers except the last in a sequence of linear (convolutional and fully connected) layers without delimiting activations are regarded as superfluous. Additionally, all layers except the last of identical type in uninterrupted sequence are regarded as superfluous, even if they do not technically conform to the aforementioned definition. The only exception to this rule is activation layers, which are allowed to directly succeed other activation layers of different kind. An example illustrating change in network topology by neural cleansing is presented in Fig. 4. Whereas this process imposes preconceived beliefs about what constitutes a well-behaving network, it continuously frees up resources which can (hypothetically) generate higher performance on the margin. The process requires a hyperparameter T, representing the (positive integer) number of generations separating consecutive cleansing operations.

3. Experiments

Evaluation of the method for CNN discovery was done in two consecutive phases: (1) *topology search*, wherein a topology was automatically constructed to perform myoelectric movement decoding by applying our EA and (2) *topology evaluation*, wherein the performance and generalizability of the single topology obtain in the search was measured. There was no overlap between data used for search phase training, search phase fitness calculation, evaluation phase training and evaluation phase testing. All parts of the experiment were performed using custom code written for and executed by Python 3.6. The TensorFlow [63] library was used to instantiate and run the genome-encoded neural networks when necessary (for fitness evaluation during the EA search phase and for training and inference during the evaluation phase).

3.1. Topology search

3.1.1. Search phase data acquisition

Movement-annotated sEMG signals were acquired from the NinaPro database [64]. From this repository the second sub-database (referred to here as NinaPro-DB2); constituted by recordings from 40 healthy volunteering subjects while performing, in addition to the resting state, 49 hand- and wrist movements, was used as the basis for generating topologies. Each recorded movement was performed for a total of 6 repetitions lasting 5 s each, separated by 3 s of rest. The obtained sEMG measurements took the form of 12 separate digital time series sampled at a rate of 2 kHz and a single synchronously acquired target signal encoding the movement class being voluntarily performed at the time of sEMG acquisition. Signals acquired from subjects 1 to 4 were used for the purpose of evolutionary search, leaving the remaining 36 subjects (5–40, henceforth referred to as *residual subjects*) for later validation (as described in section 3.2). This step assumed that the performance of a network topology evolved via evaluation on data collected from a small number of subjects is predictive of its performance on the data collected

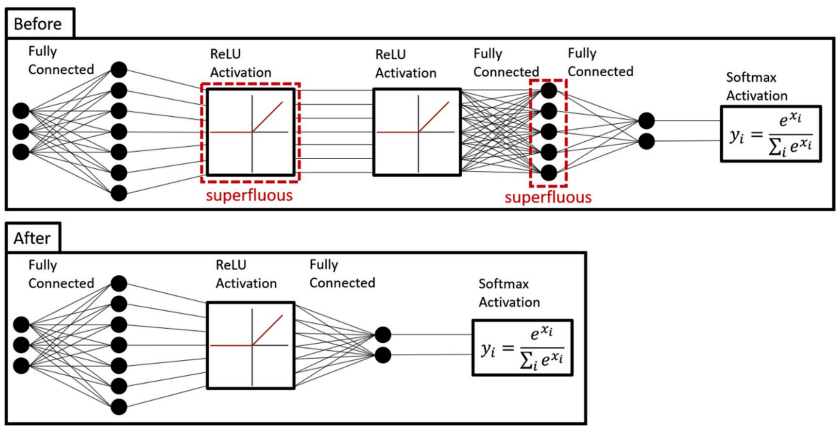


Fig. 4. The neural cleansing procedure illustrated by a fictional example. The network topologies prior to (upper) and following (lower) the procedure are equivalent in the sense that there for a configuration of learnable parameters (weights and biases) for one exists a configuration for the other that represents an identical mapping between input and output for all possible inputs. Crucially, the network topology after the operations has a smaller memory footprint and requires fewer floating-point operations per forward pass and backward pass during both training and inference.

from the remaining subjects.

3.1.2. Search phase data preprocessing

The sEMG time series were digitally filtered channel-wise, first with a 50 Hz notch filter (Q = 30) for removal of power line interference and thereafter with an 8:th order Butterworth band pass filter with a 20–380 Hz passband for suppression of high- and low frequency noise. To accelerate the EA search phase, signals were downsampled by a factor of 2 (i.e. to a new effective sampling rate of 1 kHz). A simple sliding time window technique [65] was used in order to convert the filtered time series into images processable by a CNN. This resulted in the creation of tensors of shape $L \times C \times 1$, where L is the time window length in samples and C is the number of separate channel time series (i.e. the number of electrodes; $C = 12$ for NinaPro-DB2). The last dimension represents the image colour channels; initially unitary (grayscale) but modifiable by e.g. convolutional layers. The first (vertical) and second (horizontal) dimensions correspond to time and electrode index, respectively. Each 'pixel' at this stage thus represents a raw myoelectric voltage as sampled by one electrode at one time instant. To suppress the number of computations required per individual and image during fitness evaluation, a comparatively short time window of duration 100 ms (L=100 at F = 1 kHz), with window increments of 25 ms (25% overlap) was used.

The ground truth movement class of each image-represented time window was determined via majority voting over its constituent time samples. Movement repetition was used to segment the available time windows into a training- and validation sets (per subject) as required by the fitness evaluation procedure: Images originating from movement repetitions 1 to 4 were used for training, while those from movement repetitions 5 and 6 were used for validation; i.e. as the basis for calculating fitness as described in section 2.3. As with movement class, movement repetition affiliation was determined for each image via majority voting over the samples of its time window. All images were subjected to pixel-wise normalization, i.e. by subtracting the (pixel location-specific) mean and dividing the (pixel location-specific) standard deviation; estimated exclusively from the images in the training set. Lastly, both training and validation data were balanced via the random removal of example instances (an example instance here refers to an image with corresponding class label) of overrepresented classes until the number of training and test examples of each movement class was exactly equal for all movement classes; this step ensured that the fitness of topologies was not impacted by bias towards more abundant movement classes, and furthermore reduced the computational resources required for fitness evaluation.

3.1.3. Running the search

Table 1 contains the values of the hyperparameters used by the evolutionary topology search. These values were selected *ad-hoc* via a small number of unsystematic trials and were not subject to any manual fine-tuning. Once the last evaluation step was completed, the fittest individual present in the population was extracted and used as the basis for the concluding topology evaluation. With 0.05 as the value of a_2, the lowest (worst) and highest (best) achievable values of fitness were $f_{min} = 2^{\circ/0.05} = 2^{\circ} = 1$ and $f_{max} = 2^{1/0.05} = 2^{20} = 1048576$, respectively. Based on the current state of available embedded systems [66], the default values $C_s^{max} = 100$ MB and $C_t^{max} = 100$ MFLOPS for the upper limits to inference-time memory footprint and time complexity, respectively, were used. The search was performed on a desktop computer equipped with an Nvidia Titan V GPU.

Table 1
Summary of the hyperparameters required for the evolutionary topology search and their respective values selected for the experiments.

Name	Symbol	Value
Population cardinality	C_p	250
Total number of generations	G	200
Fitness-doubling accuracy difference	a_2	5%
Mutational stability	S	50%
Mutation type probabilities	–	*add-layer*: 10% *remove-layer*: 15% *modify-layer:* 30% *modify-initializer*: 15% *modify-regularizer*: 15% *modify-optimizer*: 15%
Topology memory footprint limit	C_s^{max}	100 MB
Topology time complexity limit	C_t^{max}	100 MFLOPS
Period of neural cleansing	T	32

3.2. Topology evaluation

3.2.1. Evaluation phase data acquisition

In addition to signals acquired from the residual (5–40) subjects from NinaPro-DB2, 3 additional publicly available sEMG data sources were utilized for model evaluation:

- First sub-database of NinaPro [64] (referred to here as NinaPro-DB1).
- The sub-database of BioPatRec [67] containing 26 movement classes (referred to simply as BioPatRec).
- First (preprocessed) sub-database of the CapgMyo database [47] (referred to simply as capgMyo-DBa).

NinaPro-DB1 consisted of signals acquired from 27 subjects performing 52 movements (10 repetitions) and rest. The data was provided in the form of 10 channels sampled at $F_s = 100$ Hz. BioPatRec consisted of signals acquired from 18 subject performing 27 movements (3 repetitions) and rest. The data was provided in the form of 8 channels sampled at $F_s = 2$ kHz. CapgMyo-DBa consisted of signals acquired from 18 subjects performing 8 movements (10 repetitions). The data was provided in the form of 128 channels, arranged in a 16 × 8 grid, and sampled at $F_s = 1$ kHz.

3.2.2. Evaluation phase data preprocessing

Preprocessing of the signals originating from the residual subjects of NinaPro-DB2 was performed in a similar fashion to the signals from the original subjects (1–4) used by the evolutionary search procedure. Deviations from the previously outlined preprocessing pipeline; made in order to be able to compare results with previous work, consisted of: (1) using a window length of 200 ms (L=100 at F = 1 kHz) with 100 ms window increments (as well as the previously used 100 ms version with 25 ms increments), (2) using movement repetitions 1, 3, 4, and 6 for training and movement repetitions 2 and 5 for testing [64], and (3) not balancing images with regard to movement class abundance.

NinaPro-DB1 was preprocessed identically to NinaPro-DB2, although without any temporal filtering and exclusively using a time window of size 200 ms (L=20 at F = 100 Hz) and with increments of 100 ms. Training/testing set segmentation was once again made via repetition affiliation in accordance with previous work [64]; with movement originating from repetitions 1, 3, 4, 6, 8, 9 and 10 constituting the training set and with movements originating from repetitions 2, 5, and 7 constituting the test set.

The time series of the BioPatRec dataset were bandpass filtered, just as NinaPro-DB2, prior to channel-wise downsampling by a factor 2 in order to conform to the standard sampling rate of $F_s = 1$ kHz. Otherwise unmodified, the data was preprocessed in accordance with the convention established in previous work [31,43], i.e. samples collected during the first and last 15% of each movement repetition were discarded and movement repetition 1 was used for training, leaving repetitions 2 and 3 for accuracy assessment. To generate images, a sliding window of size 150 ms (L=150) with increments of 50 ms (33% overlap) was once again used. These images were, just as for both of the NinaPro sub-databases, subject to pixel-wise normalization.

Two distinct approaches were undertaken when applying the

evolved topology to the high-density surface recordings of the CapgMyo database. In the first approach, the measured myoelectric voltages were fed to the neural network in the form of 3-dimensional tensors of shape $L \times H \times W$ and $L \times W \times H$, respectively ('colour image' representations). Here, the height H is the electrode array vertical dimension (16 for CapgMyo-DBa) and the width W is the electrode array horizontal dimension (8 for CapgMyo-DBa). With the second approach, positional relationships between channels were ignored and the arrays of electrode measurements were fed to the neural network in the form of 3-dimensional tensors of shape $L \times W{\cdot}C \times 1$ (flattened 'grayscale image' representation). To allow for comparisons with earlier work [40,42,43], results for images corresponding to a 150 ms window duration (L=150) were computed. The time window increment was set to be equal to the time window size (0% overlap). Images originating from even movements repetitions were used for training, whereas those originating from odd repetitions were used for testing. Lastly, images were as usual linearly transformed to have (pixel-wise) zero mean and unit variance.

3.2.3. Determining performance

The topology specified by the genome of the fittest individual of the last generation was used to instantiate and subsequently optimize a single CNN movement classifier per test subject and database (input data formatted as described in the preceding section). Just as in the search-phase fitness evaluation step, the Adam algorithm with learning rate, weight initialization and weight decay specified by genome was used to train each CNN. No longer subject to as harsh incentives to keep per-subject runtime low vis-a-vis the evolutionary search phase, the number of training epochs was increased to 100 (with minibatch size 512). Once trained, the performance of each CNN was quantified by determining its classification accuracy on its respective set of test sEMG images, as defined by (4).

$$accuracy = 100\%{\cdot}\frac{number\ of\ correctly\ classified\ images}{total\ number\ of\ images} \tag{4}$$

4. Results and discussion

The cumulative execution time of the evolutionary search phase amounted to approximately 45 days. The fitness distribution of the population as it varied throughout the evolutionary search is shown in Fig. 5.

Unsurprisingly, the fastest improvements of performance were made early in the evolutionary search process, where there presumably existed many possible topology modifications with net positive impact on accuracy and hence fitness. Further along the process, when topologies had grown more intricate, most of the possible mutation sequences available to an individual were likely distinctly detrimental to fitness, leading to a much greater shortage of possible improvements. The persistent decrease in rate of fitness increase with regard to elapsed time was further caused by the increasing (mean) time complexity of topologies, increasing evaluation time required per generation. Notably, however, fitness never seems to saturate entirely, making the possibility of the process having converged close to a global optimum at the final simulated generation unlikely. The final topology, with all evolved per layer parameters, is shown in Table 2. Interestingly, the network shares many features typically observed with manually designed topologies, but moreover include some unconventional elements. The use of alternating convolutional and activation layers followed by a small fully connected subnetwork is familiar, as is the use of strided convolutions and pooling for downsampling early in the network. Surprisingly, the search resulted in a topology exclusively using filter kernels and pooling windows with extension only in the temporal image dimension. Although ReLU activations; ubiquitous in the contemporary Deep Learning literature [2], are equally available to the generative process via mutation, the topology mostly uses the traditional tanh activation function. Additional unconventional 'choices' of design include the use of a combined tanh-ReLU activation just prior to the fully connected subnetwork, and a single batch normalization layer for the entire network; just preceding the terminal classificatory FC-softmax layer pair.

Table 3 presents the classification accuracies obtained by training and subsequently using the evolved topology from Table 2 for inference on the different sources of sEMG data. When trained and subsequently evaluated on NinaPro-DB1, the evolved topology achieved a mean test accuracy of 81.4% (SD 4.0%) across the 27 subjects. The time required to train the evolved topology on data from a single subject in this dataset was 51.56 ± 7.30 s. For the 36 residual (i.e. not already used for topology search) subjects of NinaPro-DB2 the mean test accuracy amounted to 71.0% (SD 7.7%) when applied to the 'native' time window size of 100 ms and increased only slightly to 71.6% (SD 7.3%) with a window size of 200 ms. The required network training times per subject were 38.35 ± 5.60 s and 56.32 ± 4.99 s for a window size of 100 and 200 ms, respectively. Unexpectedly, the performance on NinaPro-DB1 significantly exceed that of NinaPro-DB2, in spite of the latter having

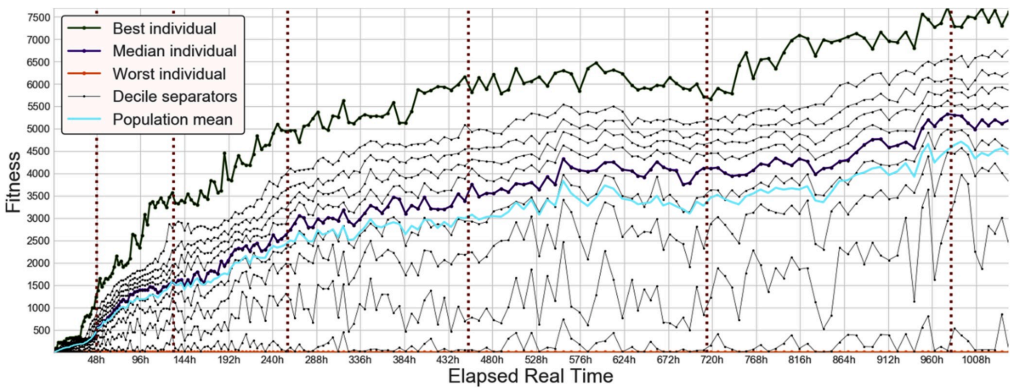


Fig. 5. The population fitness distribution as it varied during the evolutionary search. The dotted horizontal lines separate adjacent generations between which neural cleansing was performed. The time spent per generation increased, almost monotonically, throughout runtime as increasingly complex topologies arose and spread. The fitness distributions were negatively skewed for the populations of most generations, with the median individual fitness exceeding the population mean at all points in time after generation 63 (at approximately 130 h into runtime).

Table 2
Topology encoded by the genome of the fittest individual in the population of the last generation (200). The coevolved learning rate was 2.170 · 10-3. The approximated values of memory footprint and time complexity of the topology (assuming input tensors of shape 100 × 12 × 1 and 50 output neurons) are 14.7 MB and 98.3 MFLOPS, respectively.

Layer	Evolved parameter values	
0. Input	*N/A*	
1. Convolutional	number of kernels:	64
	kernel height:	9
	kernel width:	1
	vertical stride:	3
	horizontal stride:	1
	initializer std. dev.:	$4.190 \cdot 10^{-3}$
	weight decay rate:	$1.673 \cdot 10^{-4}$
2. Pooling	kind:	'max'
	window height:	8
	window width:	1
	vertical stride:	2
	horizontal stride:	1
3. Activation	kind:	'tanh'
4. Convolutional	number of kernels:	64
	kernel height:	5
	kernel width:	1
	vertical stride:	2
	horizontal stride:	1
	initializer std. dev.:	$1.537 \cdot 10^{-1}$
	weight decay rate:	$2.071 \cdot 10^{-5}$
5. Activation	kind:	'tanh'
6. Dropout	probability:	0.2093
7. Activation	kind:	'ReLU'
8. Fully Connected	output neurons:	512
	initializer std. dev.:	$2.498 \cdot 10^{-4}$
	weight decay rate:	$1.517 \cdot 10^{-7}$
9. Batch Normalization	*N/A*	
10. Activation	kind:	'tanh'
11. Fully connected	output neurons:	*number of classes*
	initializer std. dev.:	$1.343 \cdot 10^{-2}$
	weight decay rate:	$2.504 \cdot 10^{-8}$
12. Activation	kind:	'softmax'

more structural similarities (sampling rate, number of channels, etc.) with the data used for optimizing the topology than the former, as well as having slightly smaller movement set cardinality. We postulate this to be an effect, as observed in the performance of e.g. [39,43], of the general difficulty of extracting actionable movement information from raw, unsmoothed sEMG, given its apparent stochasticity, without more sophisticated signal preprocessing and/or feature extraction. With data from BioPatRec the topology evolved in the present study, albeit producing adequate results (91.4%, SD 4.6%) following a per-subject training time of 17.33 ± 2.97 s, did not surpass the 'classical' (i.e. not using Deep Learning) status quo (92.9%) [31]. On the HD-sEMG signals acquired from CapgMyo-DBa, the evolved topology reached a performance (99.3%, SD 0.6%) comparable to, although slightly worse than, the state of the art (99.7%) as reported by [42,43] after a model training duration of 38.02 ± 1.00 s per subject.

Importantly, unconditional comparisons between the performance of the example topology evolved in the present study and the Deep Learning methods of other studies are not inherently meaningful for a number of reasons: Firstly and perhaps most importantly, in contrast to earlier work, the presented topology is explicitly limited in its use of computational resources. The classifier described in the present study thus performs with a resource efficiency in compliance with available embedded platforms; practically necessary considerations which have, to the best of our knowledge, been entirely absent from previous reports on myoelectric pattern recognition based on CNNs. Secondly, more sophisticated image creation and CNN input data representation techniques, such as the spectrogram-based techniques employed by [41,43], are not available to the topology that evolved in the present study. Lastly, some specialized layer types; conjecturally instrumental in the successes of earlier studies of myoelectric pattern recognition (e.g. the locally connected layers used by [40] and the recurrent layers used by [43]), are not included in the search space of the EA. Thus, only the CNN topology presented in [39], with reported accuracy scores which were significantly outperformed by the evolved topology, can be directly compared to the topology that emerged in the present study.

5. Conclusions

This paper has presented the results of employing a novel topology selection evolutionary algorithm for the task of finding a CNN-based classifier which is (1) suitable for myoelectric pattern recognition and (2) restricted in its inference-time requirements of memory and computations. The main objective of the study was to show the feasibility of automatic approaches, sparse in their number of free design parameters, to modelling the relationship between myoelectric activity and movement information. The method was successful in creating a topology which could discriminate between the movements of some publicly available sEMG datasets almost as well as handcrafted state of the art models, despite using significantly fewer computational resources and

Table 3
Classification accuracies (averaged across subjects) achieved by the best evolved topology when evaluated on some publicly available sEMG datasets, along with the state of the art as reported by previous work incorporating CNN models. Standard deviations across subjects are presented (following ±) in cases when reported by the original publications. In the high-density database (CapgMyo-DBa), '2D representation' refers to the data with relative electrode positioning preserved (i.e. in matrix form) whereas 'flattened' refers to the same data in vectorized format (rows stacked). The column 'Reachable by EA?' indicates whether the model lies within the search space of (some hyperparameter configuration of) the presented EA.

	NinaPro [64]			BioPatRec [67]	CapgMyo-DBa [47]			Reachable by EA?
	DB1	DB2		26 movements version	2D representation		flattened	
	200 ms	100 ms	200 ms	150 ms	150 ms		150 ms	
Benchmark classifier with handcrafted feature set.	75.3% ± 5.7% (from [39])	–	75.3% ± 7.9% (from [39])	92.9% ± 4.4% (from [31])	.		99.0% ± 0.9% (from [31])	No
CNN by Atzori et al. [39]	66.6% ± 6.4%	–	60.3% ± 7.7%	–	–		–	Yes
CNN by Geng et al. [40]	77.8%	–	–	–	99.5%		–	No
Multistream-CNN by Wei et al. [42]	84.0%	–	–	–	99.7%		–	No
CNN with PCA by Zhai et al. [41]	–		78.7%	–	–		–	No
CNN-RNN by Hu et al. [43]	87.0%	–	82.2%	94.1%	99.7%		–	No
Evolved CNN (from Table 2)	81.4% ± 4.0%	71.0% ± 7.7%*	71.6% ± 7.3%*	91.4% ± 4.6%	$L \times H \times W$ 98.0% ± 1.9%	$L \times W \times H$ 98.1% ± 1.8%	99.3% ± 0.6%	Yes

*On residual subjects (i.e. 5-40)

having been generated without any manually specified design principles. Both the proposed search algorithm and the resulting topology produced here could be utilized in future work involving myoelectric pattern recognition.

As the fitness growth showed no indication of stopping during the search process, future work is needed to focus on a more ambitious search phase, with larger population size and lasting for more generations, perhaps via the use of parallelization of fitness evaluations. As the main challenge lies in the processing power required by the search and not in data storage or transfer, cloud computing could be successfully leveraged to this end. As for the search algorithm itself, its current instantiation represents a quite general formulation which could be extended in a multitude of ways in order to better accommodate the intended problem domain. The topology encoding scheme could be improved to increase the potential of candidate models by, for example, including multiple additional layer types and some parametrization of relevant signal preprocessing and image generating techniques. Similar improvements are certainly likely to be possible for other elements of the presented EA. Beyond algorithmic refinements, additional studies are needed to explore the value of the obtained results in the context of clinical adoptions: As the current study produced a CNN topology which can function with the relatively limited computational resources available on embedded platforms, evolved classifiers implemented on robotic limbs could potentially be the subject of future work. Lastly, the stability, robustness and online performance of such implementations would need to be verified empirically (and compared to the status quo) before results can be deployed widely and thereby benefit amputees.

The evolutionary approach employed in this study is constrained by some infrastructural requirements not exhibited by the handcrafted (i.e. with a static classifier topology and hyperparameter configuration) approaches to myoelectric pattern recognition which dominate the literature. Notably, automatic design of topology introduces a risk of *hyperparameter overfitting*; a problem which in the present study is eliminated by using separate data sets for topology search, training of the resulting classifier, and final evaluation. Full utilization of automatic topology discovery of the kind presented here for constructing classifiers adapted to individual users would thus require more extensive data collection compared to the status quo. Including the processing power and/or time required in order to run the search itself, the effort required in order to obtain a single topology can be substantial. Fortunately, as we have demonstrated empirically, the method is successful in creating a topology with general applicability in the problem domain, even when the search is instead conducted with a data sample shared among subjects and with a relatively parsimonious search phase. Presumably, a larger one-time investment in the form of a larger data set and a more extensive search phase, undertaken prior to deployment, could be used in order to yield a more effective topology which can subsequently be adapted to data from individual users, as would be the case with traditional, handcrafted classifier topologies.

Declaration of competing interest

None declared.

Acknowledgements

This work was supported by the Promobilia Foundation, the Crafoord Foundation, the European Commission under the DeTOP project (LEIT-ICT-24-2015, GA #687905), and the Swedish Research Council (DNR 2019–05601). The study sponsors had no role in the study design, in the collection, analysis and interpretation of data; in the writing of the manuscript; or in the decision to submit the manuscript for publication. We gratefully acknowledge the support of the NVIDIA Corporation with the donation of the Titan V GPU used for this research.

Appendix A. Layer Routines

A.1 Layer Initialization

Here follow the procedures used for selecting initial configuration of new layers after they have been created by the *add-layer* mutation.

A.1.1 Fully Connected Layer

A fully connected layer is initialized by:

1. Setting the number of output neurons to 2^n, where n is sampled from the discrete uniform distribution between 0 and 10 (i.e. log-uniformly distributed).
2. Setting the weight initializer standard deviation to 10^x, where x is sampled from the continuous uniform distribution between -3 and 0 (i.e. log-uniformly distributed).
3. Setting the weight decay rate to 10^x, where x is sampled from the continuous uniform distribution between -9 and 0 (i.e. log-uniformly distributed).

A.1.2 Convolutional Layer

A convolutional layer is initialized by:

1. Setting the number of kernels to 2^n, where n is sampled from the discrete uniform distribution between 0 and 7 (i.e. log-uniformly distributed).
2. Setting the kernel height to $2n+1$, where n is sampled from the discrete uniform distribution between 0 and 7.
3. Setting the kernel width to $2n+1$, where n is sampled from the discrete uniform distribution between 0 and 7.
4. Setting the vertical stride to n, where n is sampled from the discrete uniform distribution between 1 and the kernel height.
5. Setting the horizontal stride to n, where n is sampled from the discrete uniform distribution between 1 and the kernel width.
6. Setting the weight initializer standard deviation to 10^x, where x is sampled from the continuous uniform distribution between -3 and 0 (i.e. log-uniformely distributed).
7. Setting the weight decay rate to 10^x, where x is sampled from the continuous uniform distribution between -9 and 0 (i.e. log-uniformly distributed).

A.1.3 Pooling Layer

A pooling layer is initialized by:

1. Setting the pooling kind to ‘max’ or ‘average’ with equal probability.
2. Setting the window height to n, where n is sampled from the discrete uniform distribution between 1 and 12.
3. Setting the window width to n, where n is sampled from the discrete uniform distribution between 1 and 12.
4. Setting the vertical stride to n, where n is sampled from the discrete uniform distribution between 1 and the window height.
5. Setting the horizontal stride to n, where n is sampled from the discrete uniform distribution between 1 and the window width.

A.1.4 Activation Layer

An activation layer is initialized by setting the activation type to ‘tanh’ or ‘ReLU’ with equal probability.

A.1.5 Dropout Layer

A dropout layer is initialized by setting the dropout probability to x, where x is sampled from the continuous uniform distribution between 0 and 1.

A.1.6 Batch Normalization Layer

A batch normalization layer does not contain any tuneable hyperparameters. As a consequence, layers of this type require no specific routine for initialization.

A.2 Layer Modification

Here follow the procedures used for altering the parameters of a layer following subjection to the *modify-layer* mutation.

A.2.1 Fully Connected Layer

A fully connected layer is modified by changing the number of output neurons to 2^n, where n is sampled from the discrete uniform distribution between 0 and 10 (i.e. log-uniformly distributed).

A.2.2 Convolutional Layer

A convolutional layer is modified by randomly selecting, with equal probabilities, an action among the possibilities given by:

- Changing the number of kernels to 2^n, where n is sampled from the discrete uniform distribution between 0 and 7 (i.e. log-uniformly distributed).
- Changing the kernel height to $2n + 1$, where n is sampled from the discrete uniform distribution between k and 7, and k is the result of integer division between the vertical stride and 2.
- Changing the kernel width to $2n + 1$, where n is sampled from the discrete uniform distribution between k and 7, and k is the result of integer division between the horizontal stride and 2.
- Changing the vertical stride to n, where n is sampled from the discrete uniform distribution between 1 and the kernel height.
- Changing the horizontal stride to n, where n is sampled from the discrete uniform distribution between 1 and the kernel width.

A.2.3 Pooling Layer

A pooling layer is modified by randomly selecting, with equal probabilities, an action among the possibilities given by:

- Switching activation kind (i.e. from ‘average’ to ‘max’ or vice versa).
- Changing the window height to n, where n is sampled from the discrete uniform distribution between the vertical stride and 12.
- Changing the window width to n, where n is sampled from the discrete uniform distribution between the horizontal stride and 12.
- Changing the vertical stride to n, where n is sampled from the discrete uniform distribution between 1 and the window height.
- Changing the horizontal stride to n, where n is sampled from the discrete uniform distribution between 1 and the window width.

A.2.4 Activation Layer

A pooling layer is modified by switching activation kind (i.e. from ‘tanh’ to ‘ReLU’ or vice versa).

A.2.5 Dropout Layer

A dropout layer is modified by changing the dropout probability to x, where x is sampled from the continuous uniform distribution between 0 and 1.

A.2.6 Batch Normalization Layer

A batch normalization layer has no hyperparameters and can therefore not be subject to modification.

Appendix B. Online Complexity Estimation

B.1 Memory Footprint

To estimate inference-time memory footprint of a topology during EA runtime a simple heuristic method is used. With the approach taken in the present study, the total memory required is computed simply as the sum of the memory allocated by each individual layer. Each layer has a memory footprint contribution dependant both on the shape of its input tensor and its current parameter configuration. Layers of all types have a base memory allocation constituted by their output volume. The base memory allocation of any layer with an output volume of shape $H \times W \times C$ is equal to $4H \cdot W \cdot C$ bytes, as single-precision (32 bit) floating-point numbers (4 bytes per number) are used to represent output volume elements. Layers incorporating trainable parameter values (weights and biases), i.e. fully connected layers, convolutional layers, and batch normalization layers, allocate additional memory to accommodate these learned values as they are each represented by a single-precision floating point number. A fully connected layer with n

output neurons applied to an input size of m (where $m = H{\cdot}W{\cdot}C$ if the input is a volume of shape $H \times W \times C$) utilizes $m{\cdot}n$ weights and n biases, and thus allocates $4n{\cdot}(m+1)$ bytes of additional storage. A convolutional layer with n kernels, kernel height y, and kernel width x applied to an input volume of shape $H \times W \times C$ utilizes $n{\cdot}y{\cdot}x{\cdot}C$ weights and n biases, and thus allocates $4n{\cdot}(y{\cdot}x{\cdot}C+1)$ bytes of additional storage. A batch normalization layer applied to an input of shape $H \times W \times C$ utilizes C learned means, C learned standard deviations, C weights and C biases, and thus allocates $4{\cdot}\ 4{\cdot}\ C = 16C$ bytes of additional storage.

B.2 Time Complexity

The true time complexity of a CNN topology during inference is difficult to assess accurately, as it depends heavily on the hardware architecture upon which the program is executed [66]. For the purpose of this study, a very simple approximation is deemed sufficient. As a first step the estimate entails determining the number of floating-point operations (FLOP) required for the forward pass of one image during inference. Just as for memory footprint, an estimate of the per layer computational cost (i.e. number of FLOP) is computed and thereafter summed over the entirety of the network. Only fully connected and convolutional layers are taken into consideration, as they constitute the assumed clear majority of operations. A fully connected layer with n output neurons applied to an input size of m ($m = H{\cdot}W{\cdot}C$ if the input is a volume of shape $H \times W \times C$) performs $n{\cdot}\ m$ multiplications and $n{\cdot}m + n$ additions and thus requires $2n{\cdot}m + n$ FLOP per forward pass. A convolutional layer with n kernels, kernel height y, kernel width x, vertical stride v and horizontal stride h applied to an input volume of shape $W \times H \times C$ applies its filter kernel at $N = \frac{H}{v}\frac{W}{h}$ different input image positions (given zero-padding). At each such input image position, the convolution operation performs $y{\cdot}x{\cdot}C$ multiplications and $y{\cdot}\ x{\cdot}\ C+1$ additions (per kernel). This results in a total requirement of $N{\cdot}n{\cdot}(2x{\cdot}y{\cdot}C+1)$ FLOP. The approximation of the per-inference number of floating-point operations O_{total} by the entirety of the network (by summation over all fully connected and convolutional layers) is multiplied with the number of times inference is to be performed per second. As the first dimension H of the network input volume is assumed to correspond to time, the number of floating point operations per second (FLOPS) required by the topology is equal to $F_s{\cdot}O_{total}{\cdot}H^{-1}$, where F_s is the sampling rate of the signal(s) used to create the network input.

References

[1] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT press, Cambridge, Massachusetts, 2016.
[2] Y. LeCun, Y. Bengio, G. Hinton, "Deep learning," *Nature* 521 (2015) 436–444.
[3] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L. D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural Comput. 1 (4) (1989) 541–551.
[4] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg, L. Fei-Fei, ImageNet large scale visual recognition challenge, Int. J. Comput. Vis. 115 (2015) 211–252.
[5] ImageNet LSVRC [Online]. Available: http://image-net.org/challenges/LSVRC/. Accessed 20-3-2019.
[6] A. Krizhevsky, I. Sutskever, G. Hinton, ImageNet classification with deep convolutional neural networks, Proc. Adv. Neural Inf. Process. Syst. 25 (2) (2012).
[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., 2015.
[8] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recogn. (2016) 770–778.
[9] T. Ko, V. Peddinti, D. Povey, M.L. Seltzer, S. Khudanpur, A study on data augmentation of reverberant speech for robust speech recognition, In Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (2017) 5220–5224.
[10] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, P. Kuksa, Natural language processing (almost) from scratch, J. Mach. Learn. Res. 12 (2010) 2493–2537.
[11] I. Wallach, M. Dzamba, A. Heifets, AtomNet: a deep convolutional neural network for bioactivity prediction in structure-based drug discovery, 2015 [Online]. Available: https://arxiv.org/abs/1510.02855v1.
[12] A. Phinyomark, E. Scheme, EMG pattern recognition in the era of big data and deep learning, Big Data Cogn. Comput. 2 (3) (2018) 21.
[13] K. Mills, The basics of electromyography, J. Neurol. Neurosurg. Psychiatry 76 (2) (2005) 32–35.
[14] D. Kim, N. Lu, R. Ma, Y. Kim, R. Kim, S. Wang, J. Wu, S. Won, H. Tao, A. Islam, K. Yu, T. Kim, R. Chowdhury, M. Ying, L. Xu, M. Li, H. Chung, H. Keum, "Epidermal electronics," *Science* 333 (6044) (2011) 838–843.
[15] R. Jimenez-Fabian, O. Verlinden, Review of control algorithms for robotic ankle systems in lower-limb orthoses, prostheses, and exoskeletons, Med. Eng. Phys. 34 (4) (2012) 397–408.
[16] D. Farina, N. Jiang, H. Rehbaum, A. Holobar, B. Graimann, H. Dietl, O. Aszmann, The extraction of neural information from the surface EMG for the control of upper-limb prostheses: emerging avenues and challenges, IEEE Trans. Neural Syst. Rehabil. Eng. 22 (4) (2014) 797–809.
[17] E. Biddiss, D. Beaton, T. Chau, Consumer design priorities for upper limb prosthetics, Disabil. Rehabil. Assist. Technol. 2 (6) (2007) 346–357.
[18] C. Pylatiuk, S. Schulz, L. Döderlein, Results of an Internet survey of myoelectric prosthetic hand users, Prosthet. Orthot. Int. 31 (4) (2007) 362–370.
[19] D.J. Atkins, D.C.Y. Heard, W.H. Donovan, Epidemiologic overview of individuals with upper-limb loss and their reported research priorities, J. Prosthet. Orthot. 8 (1996) 2–11.
[20] A. Saikia, S. Mazumdar, N. Sahai, S. Paul, D. Bhatia, S. Verma, P.K. Rohilla, Recent advancements in prosthetic hand technology, J. Med. Eng. Technol. 40 (5) (2016) 255–264.
[21] J. Belter, J. Segil, A. Dollar, R. Weir, Mechanical design and performance specifications of anthropomorphic prosthetic hands: a review, J. Rehabil. Res. Dev. 50 (5) (2013) 599–618.
[22] A. Fougner, O. Stavdahl, P. Kyberd, Y. Losier, P. Parker, Control of upper limb prostheses: terminology and proportional myoelectric control-a review, IEEE Trans. Neural Syst. Rehabil. Eng. 20 (5) (2012) 663–677.
[23] C. Connolly, Prosthetic hands from touch bionics, Ind. Robot 32 (4) (2008) 290–293.
[24] M. Hakonen, H. Piitulainen, A. Visala, Current state of digital signal processing in myoelectric interfaces and related applications, Biomed. Signal Process Contr. 18 (2015) 334–359.
[25] E. Scheme, K. Englehart, Electromyogram pattern recognition for control of powered upper-limb prostheses: state of the art and challenges for clinical use, J. Rehabil. Res. Dev. 48 (6) (2011) 643–659.
[26] M. Zecca, S. Micera, M.C. Carrozza, P. Dario, Control of multifunctional prosthetic hands by processing the electromyographic signal, Crit. Rev. Biomed. Eng. 30 (4–6) (2002) 459–485.
[27] A. Phinyomark, C. Limsakul, P. Phukpattaranont, A novel feature extraction for robust EMG pattern recognition [Online]. Available: https://arxiv.org/abs/0912.3973, 2009.
[28] P. Geethanjali, K. Ray, Identification of motion from multi-channel EMG signals for control of prosthetic hand, Australas. Phys. Eng. Sci. Med. 34 (3) (2011) 419–427.
[29] C. Cipriani, C. Antfolk, M. Controzzi, G. Lundborg, B. Rosen, M. Carrozza, F. Sebelius, Online myoelectric control of a dexterous hand prosthesis by transradial amputees, IEEE Trans. Neural Syst. Rehabil. Eng. 90 (3) (2011) 260–270.
[30] C. Antfolk, C. Cipriani, M. Controzzi, M. Carrozza, G. Lundborg, B. Rosen, F. Sebelius, Using EMG for real-time prediction of joint angles of a prosthetic hand equipped with a sensory feedback system, J. Med. Biol. Eng. 30 (6) (2010) 399–406.
[31] R. Khushaba, A. Al-Timemy, A. Al-Ani, A. Al-Jumaily, A framework of temporal-spatial descriptors-based feature extraction for improving myoelectric pattern recognition, IEEE Trans. Neural Syst. Rehabil. Eng. 25 (10) (2017) 1821–1831.
[32] A. Alkan, M. Günay, Identification of emg signals using discriminant analysis and svm classifier, Expert Syst. Appl. 39 (1) (2012) 44–47.
[33] H. Huang, T. Li, C. Enz, V. Koch, J. Justiz, C. Antfolk, EMG pattern recognition using decomposition techniques for constructing multiclass classifiers, in: *Proc IEEE RAS EMBS Int. Conf. Biomed. Robot. Biomechatron.*, Singpore, 2016.
[34] X. Chen, X. Zhang, Z. Zhao, J. Yang, V. Lantz, K. Wang, Hand gesture recognition research based on surface emg sensors and 2d-accelerometers, Proc. IEEE Int. Sym. Wrbl. Co. (2017) 11–14.
[35] N. Malešević, D. Markovic, G. Kanitz, M. Controzzi, C. Cipriani, C. Antfolk, Vector Autoregressive Hierarchical Hidden Markov Models (VARHHMM) for extracting finger movements using multichannel surface EMG signals, Complexity (2018), 9728264.
[36] G. Shuman, Z. Duric, D. Barbara, J. Lin, L.H. Gerber, Improving the recognition of grips and movements of the hand using myoelectric signals, BMC Med. Inf. Decis. Making 16 (2) (2016). Article ID PMC4965724.
[37] A. Phinyomark, P. Phukpattaranont, C. Limsakul, Feature reduction and selection for emg signal classification, Expert Syst. Appl. 39 (8) (2012) 7420–7431.
[38] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Network. 2 (5) (1989) 359–366.
[39] M. Atzori, M. Cognolato, H. Müller, Deep learning with convolutional neural networks applied to electromyography data: a resource for the classification of

movements for prosthetic hands, Front. Neurorob. 10 (2016) 9, https://doi.org/10.3389/fnbot.2016.00009.
[40] W. Geng, Y. Du, W. Jin, W. Wei, Y. Hu, J. Li, Gesture recognition by instantaneous surface EMG images, Sci. Rep. 15 (6) (2016) art. no. 36571.
[41] X. Zhai, B. Jelfs, R.H.M. Chan, C. Tin, Self-recalibrating surface EMG pattern recognition for neuroprosthesis control based on convolutional neural network, Front. Neurosci. 11 (2017) 379, https://doi.org/10.3389/fnins.2017.00379.
[42] W. Wei, Y. Wong, Y. Du, Y. Hu, M. Kankanhalli, W. Geng, A multi-stream convolutional neural network for sEMG-based gesture recognition in muscle-computer interface, Pattern Recogn. Lett. 119 (2017) 131–138.
[43] Y. Hu, Y. Wong, W. Wei, Y. Du, M. Kankanhalli, W. Geng, A novel attention-based hybrid CNN-RNN architecture for sEMG-based gesture recognition, PloS One 13 (2018), https://doi.org/10.1371/journal.pone.0206049.
[44] K. Park, S. Lee, Movement intention decoding based on deep learning for multiuser myoelectric interfaces, in: Proceedings of BCI, Gangwon Province, South Korea, 2016.
[45] M.Z. ur Rehman, S.O. Gilani, A. Waris, I.K. Niazi, G.G. Slabaugh, D. Farina, E. N. Kamavuako, Stacked sparse autoencoders for EMG-based classification of hand motions: a comparative multi day analyses between surface and intramuscular EMG, Appl. Sci. 8 (2018) 7, https://doi.org/10.3390/app8071126.
[46] H. Shim, H. An, S. Lee, E. Lee, H. Min, S. Lee, EMG pattern classification by split and merge deep belief network, Symmetry 8 (2016) 12, https://doi.org/10.3390/sym8120148.
[47] Y. Du, W. Jin, W. Wei, Y. Hu, W. Geng, Surface EMG-based inter-session gesture recognition enhanced by deep domain adaptation, Sensors 17 (2017) 3, https://doi.org/10.3390/s17030458.
[48] A.E. Olsson, P. Sager, E. Andersson, A. Björkman, N. Malešević, C. Antfolk, Extraction of multi-labelled movement information from the raw HD-sEMG image with time-domain depth, Sci. Rep. 9 (1) (2019), 7244, https://doi.org/10.1038/s41598-019-43676-8.
[49] A. Ameri, M.A. Akhaee, E. Scheme, K. Englehart, Regression convolutional neural network for improved simultaneous EMG control, J. Neural. Eng. 16 (3) (2019).
[50] M. Rojas-Martnex, M. Mañanas, J. Alonso, High-density surface EMG maps from upper-arm and forearm muscles, J. NeuroEng. Rehabil. 9 (2012) 85, https://doi.org/10.1186/1743-0003-9-85.
[51] C. Cortes, X. Gonzalvo, V. Kuznetsov, M. Mohri, S. Yang, AdaNet: adaptive structural learning of artificial neural networks, Proceedings of the ICML 70 (2017) 874–883.
[52] E. Real, S. Moore, A. Selle, S. Saxena, Y. Suematsu, J. Tan, Q. Le, A. Kurakin, Large-scale evolution of image classifiers, Proc. ICML 70 (2017) 2902–2911.
[53] R. Miikkulainen, J. Liang, E. Meyerson, A. Rawal, D. Fink, O. Francon, B. Raju, H. Shahrzad, A. Navruzyan, N. Duffy, B. Hodjat, Evolving deep neural networks, in: Artificial Intelligence in the Age of Neural Networks and Brain Computing, Academic Press, Cambridge, Massachusetts, United States, 2018, pp. 293–312.
[54] J. Snoek, H. Larochelle, R.P. Adams, Practical bayesian optimization of machine learning algorithms, In Proc. Adv. Neural Inf. Process. Syst. 2 (2012) 2951–2959.
[55] B. Zoph, Q. Le, Neural architecture search with reinforcement learning [Online]. Available: https://arxiv.org/abs/1611.01578, 2016.
[56] S. Han, J. Pool, J. Tran, W.J. Dally, Learning both weights and connections for efficient neural network, In Proceedings of NeurIPS 1 (2015) 1135–1143.
[57] K.O. Stanley, R. Miikkulainen, Efficient reinforcement learning through evolving neural network topologies, Proc. GECCO (2002) 569–577.
[58] T. Bäck, Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms, Oxford University Press, New York, USA, 1996.
[59] G. Kanitz, C. Antfolk, C. Cipriani, F. Sebelius, M. Carrozza, Decoding of individuated finger movements using surface EMG and input optimization applying a genetic algorithm, in: *Proceedings of EMBC*, Boston, USA, 2011.
[60] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (2014) 1929–1958.
[61] S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift [Online]. Available: https://arxiv.org/abs/1502.03167v3, 2015.
[62] D. Kingma, J. Ba, Adam: a method for stochastic optimization [Online]. Available: https://arxiv.org/abs/1412.6980v9, 2014.
[63] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D.G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, X. Zheng, TensorFlow: a system for large-scale machine learning, in: Proceedings of the USENIX Conference on Operating Systems Design and Implementation, 2016.
[64] M. Atzori, A. Gijsberts, C. Castellini, B. Caputo, A.G. Mittaz Hager, S. Elsig, G. Giatsidis, F. Bassetto, H. Müller, Electromyography data for non-invasive naturally-controlled robotic hand prostheses, Sci. Data 1 (2014), https://doi.org/10.1038/sdata.2014.53.
[65] L. Smith, L.J. Hargrove, B. Lock, T. Kuiken, Determining the optimal window length for pattern recognition-based myoelectric control, IEEE Trans. Neural Syst. Rehabil. Eng. 19 (2) (2011) 186–192.
[66] B. Moons, D. Bankman, M. Verhelst, Embedded Deep Learning: Algorithms, Architectures and Circuits for Always-On Neural Network Processing", Springer, Cham, Switzerland, 2019.
[67] M. Ortiz-Catalan, R. Brånemark, B. Håkansson, BioPatRec: a modular research platform for the control of artificial limbs based on pattern recognition algorithms, Source Code Biol. Med. 8 (2013) 11, https://doi.org/10.1186/1751-0473-8-11.

Paper III

Learning regularized representations of categorically labelled surface EMG enables simultaneous and proportional myoelectric control

A. E. Olsson, N. Malešević, A. Björkman, C. Antfolk

Published in: Journal of NeuroEngineering and Rehabilitation 18, 35 (2021)

Olsson *et al. J NeuroEngineering Rehabil (2021) 18:35*
https://doi.org/10.1186/s12984-021-00832-4

Journal of NeuroEngineering and Rehabilitation

RESEARCH Open Access

Learning regularized representations of categorically labelled surface EMG enables simultaneous and proportional myoelectric control

Alexander E. Olsson[1*], Nebojša Malešević[1], Anders Björkman[2,3] and Christian Antfolk[1*]

Abstract

Background: Processing the surface electromyogram (sEMG) to decode movement intent is a promising approach for natural control of upper extremity prostheses. To this end, this paper introduces and evaluates a new framework which allows for simultaneous and proportional myoelectric control over multiple degrees of freedom (DoFs) in real-time. The framework uses multitask neural networks and domain-informed regularization in order to automatically find nonlinear mappings from the forearm sEMG envelope to multivariate and continuous encodings of concurrent hand- and wrist kinematics, despite only requiring categorical movement instruction stimuli signals for calibration.

Methods: Forearm sEMG with 8 channels was collected from healthy human subjects (N = 20) and used to calibrate two myoelectric control interfaces, each with two output DoFs. The interfaces were built from (I) the proposed framework, termed *Myoelectric Representation Learning* (MRL), and, to allow for comparisons, from (II) a standard pattern recognition framework based on Linear Discriminant Analysis (LDA). The online performances of both interfaces were assessed with a Fitts's law type test generating 5 quantitative performance metrics. The temporal stabilities of the interfaces were evaluated by conducting identical tests without recalibration 7 days after the initial experiment session.

Results: Metric-wise two-way repeated measures ANOVA with factors method (MRL vs LDA) and session (day 1 vs day 7) revealed a significant ($p < 0.05$) advantage for MRL over LDA in 5 out of 5 performance metrics, with metric-wise effect sizes (Cohen's d) separating MRL from LDA ranging from $|d| = 0.62$ to $|d| = 1.13$. No significant effect on any metric was detected for neither session nor interaction between method and session, indicating that none of the methods deteriorated significantly in control efficacy during one week of intermission.

Conclusions: The results suggest that MRL is able to successfully generate stable mappings from EMG to kinematics, thereby enabling myoelectric control with real-time performance superior to that of the current commercial standard for pattern recognition (as represented by LDA). It is thus postulated that the presented MRL approach can be of practical utility for muscle-computer interfaces.

Keywords: Electromyography, Prosthetic control, Online performance, Regression, Deep learning, Representation learning, Regularization, Multitask learning

*Correspondence: alexander.olsson@bme.lth.se; christian.antfolk@bme.lth.se
[1] Department of Biomedical Engineering, Faculty of Engineering, Lund University, Lund, Sweden
Full list of author information is available at the end of the article

Background

Muscle-computer interfaces (MCIs) have found use in a broad range of clinical and biotechnical domains

[1]. Most salient within the category of clinical applications is perhaps the field of hand- and wrist prosthetics, where myoelectrically controlled prostheses have been part of clinical routine since the 1960s [2]. In this application, electromyography (EMG) signals are processed by an MCI and transformed into movement commands intended to modulate the behaviour of a powered actuator, i.e. a robotic replacement limb. The prototypical system [3] designed to this end utilizes a sparse setup of surface EMG (sEMG) electrodes which measure the activities of a single antagonistic muscle pair located superficially in the residual limb of the amputee. The difference in some measure of intensity (e.g. signal magnitude) between the sEMG signals from the pair can thereafter be mapped directly to the force driving a single motorized degree of freedom (DoF) which is typically instantiated as the grasp aperture of a hand-replacing gripper. Within this framework, the additional DoFs possessed by multifunctional prostheses (which have recently become more available to hand- and arm amputees [4]) must be controlled sequentially by use of auxiliary protocols, e.g. based on co-contraction [5] or non-EMG inputs [6], for DoF switching. The enduring preponderance of this direct control framework can be understood in light of the robustness brought about by the relative simplicity of the relevant hard- and software, as well as the ease with which the intensity of contraction of a single muscle group can be controlled volitionally. However, disadvantages such as limited dexterity, lack of intuitiveness, and an associated cognitive burden have been observed among users [7]; these are thought to be among the main reasons for the high abandonment rates by which devices controlled in this way are afflicted [8].

The divide that separates the direct control paradigm from advances seen in mechatronics has for a time spurred research into potential alternatives. A noteworthy candidate to this end is the use of myoelectric pattern recognition [9–11]—a class of methods which formulates the control problem as one of supervised machine learning. Within this framework, example segments $\boldsymbol{X}$ of a multichannel sEMG time series (typically acquired from the forearm) or more information-dense features [12] of the same, are, together with encodings of co-occurring movements $\boldsymbol{y}$, fed to a machine learning algorithm which generates a computable function $f_{\boldsymbol{\theta}}$. This learned function represents an approximate mapping between sEMG and movement and is typically derived by selecting the free parameters $\boldsymbol{\theta}$ such that $f_{\boldsymbol{\theta}}$ minimizes some loss metric $\sum_t \mathcal{L}(\mathbf{y_t}, \widehat{\boldsymbol{y}}_{\boldsymbol{t}}, \boldsymbol{\theta})$, where $\boldsymbol{X}_t$ and $\boldsymbol{y}_t$ are the sEMG segment and (a numeric encoding of) the concurrent movement, respectively, at time t, and $\widehat{\boldsymbol{y}}_t = f_{\boldsymbol{\theta}}(\boldsymbol{X}_t)$ is (a numeric encoding of) the inferred movement. Following such initial calibration, $f_{\boldsymbol{\theta}}$ can be used to process previously unseen segments by recognizing movement-specific sEMG patterns; an MCI based on pattern recognition can thus be understood as a form of gesture recognition system.

The contemporary engineering research literature shows no signs of scarcity when surveyed for approaches based on pattern recognition aiming to accommodate the mechanical sophistication of available robotic limbs. Algorithms from the broader machine learning discipline such as linear discriminant analysis [13]; support vector machines [14]; hidden Markov models [15]; and decision trees/random forests [16] have, among several others, been applied for this purpose; such methods have at times reached impressive classification accuracies of more than 95% for movement class sets with cardinalities exceeding 10 [17]. As in most other technical pursuits in which statistical inference plays a part, Deep Learning [18] in the form of, for example, convolutional neural networks (e.g. [19–23]) and recurrent neural networks (e.g. [24, 25]), has recently found widespread use in myoelectric control research [26] and has frequently attained exceptional accuracy scores. Unlike their 'classical' machine learning counterparts, such methods avert the need for manual feature engineering via their ability to gainfully operate directly on raw sEMG, but are often hampered by a need for time-consuming hyperparameter tuning; large datasets; and/or requirement on computational resources infeasible for embedded systems [27].

Independent of the minutiae of any specific algorithm, the improvements over the industrial and clinical status quo made possible by pattern recognition are quite apparent. Importantly, use of pattern recognition is congruent with complete naturalness of control: The task of mapping a detected movement attempt to a movement command corresponding to the very same movement is trivial, thus enabling an intuitive form of steering. Similarly, multiarticulate control can be realized either implicitly, by detecting separate multiarticulate movements and/or grasps as individual classes, or explicitly, by detecting each DoF separately using multi-output versions of pattern recognition [22, 28–30]. In spite of such alluring promises, the fact remains that remarkably few implementations of pattern recognition have so far been deployed at scale in the daily life of amputees [31].

Conjecturally, one of the main obstacles separating myoelectric pattern recognition from widespread adoption within prosthetics relates to the phenomenon of drift in the data-generating distribution $P(\boldsymbol{X}|\boldsymbol{y})$ from which sEMG is sampled [32]. Stated succinctly, the statistical relationship connecting measured myoelectric activity $\boldsymbol{X}$ to movement $\boldsymbol{y}$ is not necessarily identical to the relationship which was valid at the time of calibration data acquisition, making the problem a specific instance

of model overfitting. Variations in electrode positions; skin conductivity; limb placement and load; and fatigue are all examples of mechanisms which modulate the characteristics of the acquired sEMG [32], making the learned mapping $f_\theta(\boldsymbol{X}) = \hat{\boldsymbol{y}}$ obsolete and thus degrading MCI performance over time [33]. Drift of this kind has in the past been mitigated either by including calibration data from a varied set of recording circumstances (although this approach has limitations regarding scalability [10]) or by using adaptive control strategies [34]. As will be argued in this paper, a complementary strategy is to develop methods which yield more generalizable mappings from sEMG to movement via regularization.

In addition to problems of robustness and stability of the aforementioned kind, one drawback of straightforwardly applying pattern recognition relates to proportionality of control. To make effective use of a prosthesis it is practical, and perhaps even necessary, to be able to not only transmit what movement to perform, but also to transmit information of the desired force and velocity—a capability not granted by basic pattern recognition. A naïve solution is to reformulate the classification problem as one of direct regression (of kinetics and/or kinematics), as is certainly notionally consistent [35]. However, at some point this requires ground truth measurements of relevant regressands, which in principle are impossible to acquire from prosthesis users. One way to circumvent this anatomical limitation has been the use of mirrored training [36], where sEMG from the amputation stump, collected during mediolaterally mirrored movements, is used to infer the kinematics of the contralateral, intact limb. Regression has also been realized by using continuous visual movement instruction stimuli as regressand [21], which requires the subject to manually vary the intensity of muscle contraction during acquisition of calibration data. Regardless of method, proportional interfaces have been observed to lead to higher levels of user adaptation [37], potentially due to their greater resemblance to natural motor control.

An alternative way of extending myoelectric pattern recognition into the continuous domain, that does not require continuous target measurements, is to leverage the fact that aggregated sEMG activity can be modulated volitionally, and thusly estimate movement class and intensity of contraction separately. This approach, which has been applied both in previous laboratory studies (e.g. [38–40]) and commercially [41], use a classifier to determine what gesture is to be performed. Following classification, the detected gesture is performed with velocity directly proportional to either (I) the concurrently estimated force of contraction (with e.g. instantaneous sEMG magnitude as proxy), or to (II) some monotonously increasing function thereof. Such functions can be tuned automatically and independently for each detectable movement, thereby accounting for systematic differences in intensity between movement classes [40]. Albeit uncomplicated and demonstrably effective, these strategies can be understood as problematic for a number of reasons. Firstly, there is no guarantee that the pattern associations learned during model calibration will be generalizable to all intensities of contraction [10], and thus some sEMG patterns might inadvertently be classified as patterns cooccurring with other movement classes. Such mistakes can plausibly lead to an MCI output perceived as erratic by the user. Secondly, proportionality mediated in this way is not *simultaneous* over all available DoFs, as only a single dimension of proportional information (i.e. the globally estimated intensity of myoelectric activity) is available.

In addition to developments in pattern recognition, studies of methods which are not directly based on regression or classification have demonstrated the potential of several alternative paths towards natural, simultaneous, and proportional myoelectric control. Multisite intramuscular EMG (iEMG), which can measure motor unit action potentials directly [42], has been investigated as a mechanism for direct control, and has furthermore been shown to possess functional advantages when compared to proportional pattern recognition [43]. Weakly supervised autoencoding has shown promising results in unlabelled separation of underlying sEMG signal components which can be mapped to kinematics directly [44]. Nonnegative matrix factorization has been used [45] to extract multiple simultaneous DoFs separately from rectified and filtered sEMG while retaining their respective proportionalities. Techniques for deconvoluting high-density sEMG based on models informed by neuromuscular physiology have successfully been applied towards the same end [46]. Although at the cutting edge of electrophysiology, such approaches have, possibly due to advanced modes of signal acquisition, so far mostly been constrained to the laboratory environment.

In order to aid in the pursuit of practical MCIs and to alleviate the limitations of available methods, this paper introduces a new set of methods aimed at achieving intuitive, proportional, and simultaneous myoelectric control. Concretely, the framework is constituted by a computationally lightweight neural network topology with a compatible optimization procedure, all described in detail in the Methods section. In contrast to previous frameworks based on pattern recognition, the proposed combination of techniques operates to learn nonlinear mappings from forearm sEMG to continuous and multivariate encodings of hand- and wrist kinematics, despite only being calibrated with sEMG signals labelled with categorical movement instruction stimuli. This affords the framework the

advantage of regression-based approaches (i.e. proportionality) but requiring neither kinematic ground truth data nor complicated recording protocols. Additionally, by incorporating a multi-task learning formulation of the kinematic inference problem, the framework implicitly allows for independent and simultaneous control of all considered DoFs. Due to its reliance on signal representations [47] arising from supervised learning with regularizing constraints, the novel framework is referred to as myoelectric representation learning (MRL). To demonstrate the viability of MRL and to quantify differences in performance compared to the current commercial standard for pattern recognition, this paper includes experiments in which test subjects were tested for efficacy of control when using (I) MRL, and (II) pattern recognition as represented by linear discriminant analysis (LDA) [40], to perform a virtual Fitts's law [48] type test. Furthermore, to quantify temporal deterioration of myocontrol quality, the performances of both methods were reassessed after 7 days of intermission. Interestingly, distributed representations learned by the MRL model seem insensitive to small drifts in the data-generating distribution over time, leading to a stable interface across the two usage sessions.

Methods

Myoelectric representation learning

In accordance with existing pattern recognition frameworks for myoelectric control via supervised learning, the proposed MRL system operates in two modes: calibration and inference. During calibration, an adaptive model is trained to approximate a mapping from sEMG to concurrent movement intent. During inference, the calibrated model is used to regress kinematics from previously unseen sEMG samples in real-time. Just as is the case with many earlier myoelectric decoding systems, the MRL system comprises a preprocessing step followed by a multi-layered feedforward artificial neural network (ANN) model. To adapt the model to the task at hand, the system requires user-specific calibration data in the form of:

1 $\boldsymbol{X}^{cal} = \left[\boldsymbol{x}_1^{cal}, \cdots, \boldsymbol{x}_t^{cal}, \cdots, \boldsymbol{x}_T^{cal}\right]$, where $\boldsymbol{x}_t^{cal} \in F_b^I$ are raw sEMG voltages at time t, F_b is the set of all floating-point numbers representable by b bits, I is the number of sEMG channels, and T is the number of calibration samples.
2 $\boldsymbol{Y}^{cal} = \left[\boldsymbol{y}_1^{cal}, \cdots, \boldsymbol{y}_t^{cal}, \cdots, \boldsymbol{y}_T^{cal}\right]$, where $\boldsymbol{y}_t^{cal} \in \{-1, 0, 1\}^J$ is a ternary DoF-wise categorical encoding of the movement intent of the subject at time t, and J is the number of DoFs which the system is intended to control.

Following calibration, the system can infer DoF-wise continuous output $\widehat{\boldsymbol{y}} \in F_b^J$, corresponding to concurrent kinematics, from any time slice $\boldsymbol{x} \in F_b^I$ of a provided sEMG time series. During inference time, the system provides an output control signal with update rate identical to that with which the input sEMG is provided.

Two principal modifications distinguish the properties of the ANN models employed here from those of previous applications of Deep Learning for the purpose of decoding myoelectric signals. The first modification, to which the network topology itself is subject, can be viewed through the prism of hard parameter sharing as known from the literature on multitask learning [49]. This modification, which is elaborated upon in the section titled 'Neural Network Topology', is the basis for the simultaneity of control enabled by calibrated models. The second modification, which involves the optimization procedure and the appertaining loss function which selects the free parameters of any instantiation of the topology, can be viewed through the prism of contractive regularization, as known from the study of deep autoencoders [50]. This technique, which has been reformulated for the intended purpose and is presented in the section titled 'Calibration', allows for proportionality of control when deploying calibrated models.

Preprocessing

Prior to instantiation and optimization of the neural model, the raw I-channel sEMG signals $\boldsymbol{X}^{cal} \in F_b^{I\times T}$ of the calibration data set is subject to envelope extraction, rescaling, clipping, and nonlinear transformation: Initially, channel-wise signal envelopes are extracted from $\boldsymbol{X}^{cal}$ by full-wave rectification and digital LTI lowpass filtering with a gain-free causal FIR filter (moving average) with length W samples. The resulting (unbounded and nonnegative) envelopes $\boldsymbol{E}^u \in F_b^{I\times T}$ are thereafter linearly rescaled elementwise via (1).

$$E_{i,l}^r \leftarrow \frac{E_{i,l}^u - p_i^{1\%}}{p_i^{99\%} - p_i^{1\%}} \tag{1}$$

Here, $p_i^{1\%}$ and $p_i^{99\%}$ are the 1st and 99th percentile, respectively, of the recorded voltages of the i th sEMG channel envelope across all T samples of $\boldsymbol{E}^u$. The resulting rescaled signals $\boldsymbol{E}^r \in F_b^{I\times T}$ are subject to clipping and transformed by the element-wise square root operator as shown in (2).

$$E_{i,l}^{cal} \leftarrow \sqrt{max(0, min(1, E_{i,l}^r))} \tag{2}$$

These steps: (I) ensure that all elements of the obtained matrix $\boldsymbol{E}^{cal} = \left[\boldsymbol{e}_1^{cal} \cdots \boldsymbol{e}_t^{cal} \cdots \boldsymbol{e}_T^{cal}\right] \in F_b^{I \times T}$ (which are to be used as ANN inputs) are constrained to the interval [0,1]; (II) mitigate the influence of outlier samples in $\boldsymbol{X}^{cal}$; and (III) provide implicit threshold values under which the envelopes are taken to equal zero. The square root operator is included to bias resolution towards high levels of muscle contraction by smoothing variations in signal envelopes at values close to the observed maximum. Due to the lack of time shifts introduced in the process of generating $\boldsymbol{E}^{cal}$ from $\boldsymbol{X}^{cal}$, the columns (i.e. time points) of the ground truth movement intent matrix $\boldsymbol{Y}^{cal}$ are synchronous with those of $\boldsymbol{E}^{cal}$.

When the MRL system operates in inference mode, acquired sEMG samples are processed in an identical manner by utilizing online filtering and rescaling with the calibration data statistics $\boldsymbol{p}^{1\%} = \left[p_1^{1\%}, \cdots, p_i^{1\%}, \cdots, p_I^{1\%}\right]$ and $\boldsymbol{p}^{99\%} = \left[p_1^{99\%}, \cdots, p_i^{99\%}, \cdots, p_I^{99\%}\right]$ as per Eqs. (1) and (2).

Neural network topology

A network topology of the kind described in this section is depicted in Fig. 1. The ANN model takes as input an sEMG envelope time slice $\boldsymbol{e} = [e_1 \cdots e_i \cdots e_I]^{\boldsymbol{T}}$, obtained as described above, and provides a numeric representation of inferred concurrent kinematics $f(\boldsymbol{e}) = \widehat{\boldsymbol{y}} \in \left[\widehat{y}_1 \cdots \widehat{y}_j \cdots \widehat{y}_J\right]^{\boldsymbol{T}}$ as output. Although the ANN models of the current study are instantiated and calibrated end-to-end (as will be described in the next section), the mappings they represent can naturally be understood as processes constituted by two elements in succession: encoding followed by decoding. Initially, $\mathbf{e}$ is mapped to the input layer which in turn is fed through N blocks, the sequence of which constitute the encoder network said to be performing the function $H(\cdot)$. (The depth N of the decoder network is a hyperparameter to be selected prior to network instantiation.) Internally, each such encoder block is constituted by a fully connected layer [18] followed by a leaky rectifier linear unit (ReLU) activation layer [51], whose output in turn is subject to layer normalization [52] without learnable parameters. The numbers of output neurons of the first fully connected block of the encoder network is here set to equal 2^K, where $K \geq N$ is a model hyperparameter. The number of output neurons for each consecutive encoder block is set to equal half of that of its predecessor, and the number of output neurons at the last encoder block thus

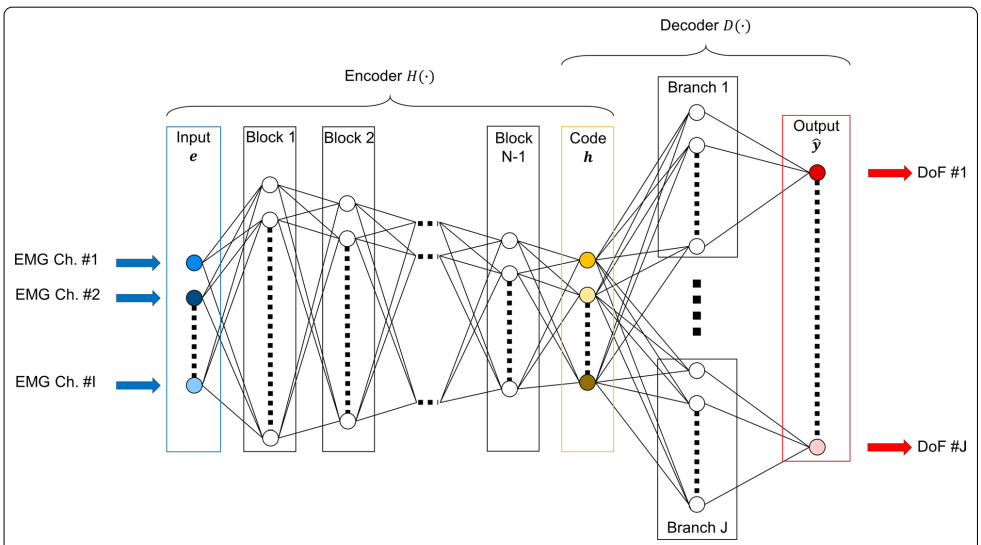


Fig. 1 Schematic overview of the regression feedforward neural network topology central to the MRL framework. A time slice $\boldsymbol{e}$ of the signal envelopes of I-channel sEMG at time t is fed through an *encoder* network, constituted by N fully connected blocks, which transforms $\boldsymbol{e}$ into an alternative *representation*, i.e. code, $\boldsymbol{h}$. A set of J *decoder* networks, each constituted by a single hidden block, decode this representation to independently estimate the activations of J DoFs, interpreted collectively as the proportional and simultaneous output command $\widehat{\boldsymbol{y}}$ corresponding to the movement intent of the user at time t

equals 2^{K-N+1}. This output of this last encoder block, referred to as the code $H(\boldsymbol{e}) = \boldsymbol{h} \in F_b^{K-N+1}$, is subsequently fed into a set of parallel decoder sub-networks, collectively said to be performing the function $D(\cdot)$. These decoder sub-network, each associated with one of the J decodable DoFs, are constituted by a single fully connected hidden block (again incorporating leaky ReLU activations and parameter-free layer normalization) of output size 2^S (S being a hyperparameter) and a terminal fully connected layer with one output neuron with linear activation function. The resulting concatenation of values $\{\widehat{y}_1, \ldots, \widehat{y}_J\}$, each of which is the output of one of the decoder networks, constitute the network output, i.e. $f(\boldsymbol{e}) = D(H(\boldsymbol{e})) = D(\boldsymbol{h}) = \widehat{\boldsymbol{y}} = \left[\widehat{y}_1, \cdots, \widehat{y}_j, \cdots, \widehat{y}_J\right]^T$, where $\widehat{y}_j$ is to be interpreted as the inferred 'intensity' of movement intent for the j th kinematic output DoF.

In previous work aimed at achieving simultaneous (i.e. multi-DoF) myoelectric control via the use of regression ANNs, a distinction is sometimes made between the use of shared models and dedicated models [36, 45]. Shared models here refer to a type of network where each DoF to be inferred is represented as a single neuron in the output layer. Dedicated models, in contrast, use one separate network per DoF, each with a single output neuron. Within this framework, it has typically been found that dedicated models outperform shared models in tasks of multivariate kinematic regression [53]. The novel ANN topology used in the current study can be construed as a hybrid between these two extremes: The encoder $H(\cdot)$ is shared between the DoFs and the decoder $D(\cdot)$ is formed by dedicated branch networks.

Use of an initial shared network followed by multiple task-specific subnetworks, sometimes referred to as hard parameter sharing [49], has previously been studied in the context of multitask learning, where the simultaneous estimation of multiple related target variables often results in better performance for all estimations individually [54]. It is thus hypothesised that the alternative presented here is an improvement over the dedicated model approach: The encoder can learn to transform sEMG into a more useful signal representation $\boldsymbol{h}$ for the purpose of decoding all DoFs, while at the same time allowing for the advantages of using dedicated subnetworks to infer the activation strengths of all DoFs separately.

Calibration

The calibration of the MRL neural network topology previously presented, i.e. the tuning of model parameters (weights $\boldsymbol{W}$ and biases $\boldsymbol{b}$) for a given user of the system, is specified by two components: (I) a differentiable loss metric $\mathcal{L} = \mathcal{L}(\boldsymbol{E}^{cal}, \boldsymbol{Y}^{cal}, \boldsymbol{\theta})$, where $\boldsymbol{\theta} = \{\boldsymbol{W}, \boldsymbol{b}\}$ is the complete set of free model parameters, and (II) an iterative, gradient descent procedure for its minimization with respect to the parameter values constituting $\boldsymbol{\theta}$.

The loss metric $\mathcal{L}$, which is to be minimized, is given below in (3) and is defined as a weighted sum of the two loss functions $\mathcal{L}_i$ and $\mathcal{L}_c$.

$$\mathcal{L} = \mathcal{L}_i + \alpha_c \mathcal{L}_c \tag{3}$$

The value of $\alpha_c \in F_b$ is a hyperparameter representing the relative weighting of the two sources of error. The first term $\mathcal{L}_i$ in the definition above is referred to as the inference loss and is given in (4).

$$\mathcal{L}_i\left(\boldsymbol{E}^{cal}, \boldsymbol{Y}^{cal}, \boldsymbol{\theta}\right) = \frac{1}{T}\sum_{t}^{T} \|\boldsymbol{y}_{\boldsymbol{t}}^{cal} - \widehat{\boldsymbol{y}}_{\boldsymbol{t}}\|_1 \tag{4}$$

Here $\|\cdot\|_1$ denotes the L_1 vector norm, i.e. $\|\boldsymbol{v}\|_1 = \sum_i |v_i|$. This loss term, which is functionally identical to DoF-wise mean absolute error, quantifies the discrepancy between ground truth movement intent $\boldsymbol{y}_{\boldsymbol{t}}^{cal}$ and the corresponding value $\widehat{\boldsymbol{y}}_{\boldsymbol{t}} = G(H(\boldsymbol{e}_t^{cal}))$ inferred by the ANN model, averaged across all calibration example instances. Importantly, in contrast to the more commonly used mean square error of linear regression fame, $\mathcal{L}_i$ penalizes DoF-wise distance from estimate to target linearly. When deployed in conjunction with the topology presented above, $\mathcal{L}_i$ can additionally be understood as a form of regularization in and of itself: In order to achieve a low value of $\mathcal{L}_i$, the model needs to accurately infer the activation of all J DoFs simultaneously. As has been observed for multitask learning models in general [54], multiple related sub-tasks of this kind impose regularizing constraints onto each other, impacting which kinds of data representations are likely to arise throughout the network and leading to a synergistic effect whereby the performance on every sub-task (i.e. the estimation quality for every individual DoF) is expected to improve.

Albeit clearly a necessary condition for desirable model behaviour, a low value of $\mathcal{L}_i$ is not a sufficient goal of the optimization procedure in the context of kinematic estimation. To realize this fact, one can consider the case of a model which has learned to infer only categorical output (i.e. $\widehat{\boldsymbol{y}} \in \{-1, 0, 1\}^J$) that perfectly matches the provided categorical targets in $\boldsymbol{Y}^{cal}$ (e.g. the model in Fig. 2b). Furthermore, assume this model performs well only with input sEMG envelopes generated at intensities of muscle contraction identical to those employed during the collection of the calibration data $\boldsymbol{E}^{cal}$. Such a model, although clearly deficient for the task at hand, would achieve very low values of $\mathcal{L}_i$. To generate calibrated models which in addition enable the previously stated goal of proportionality of control, as well as achieve adequate performance with

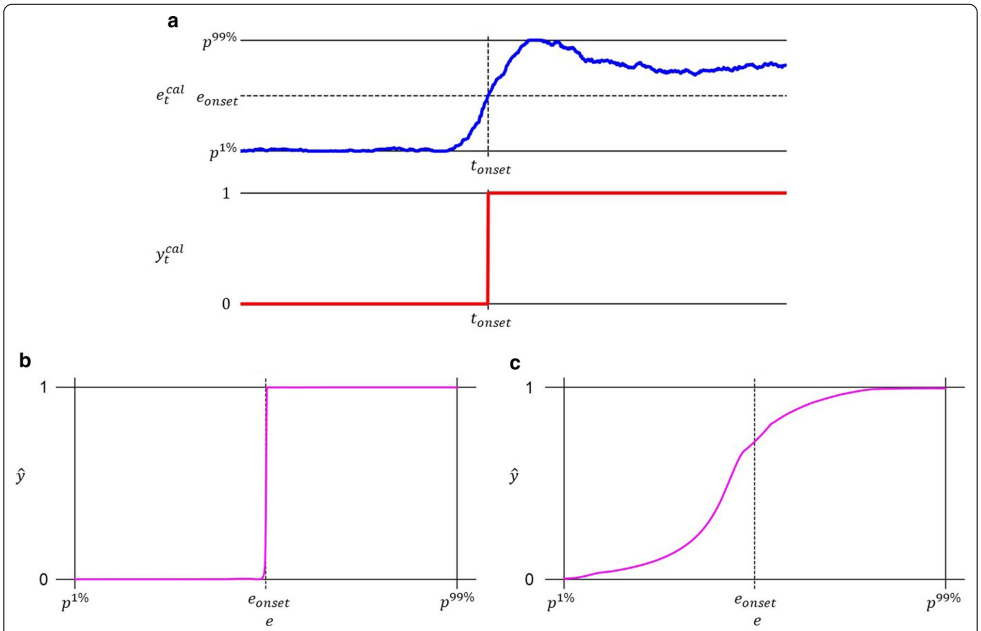


Fig. 2 The effect of contractive regularization on proportionality of control. The effect is illustrated by a simplified example with univariate regressor $\boldsymbol{e} = e$ and regressand $\boldsymbol{y} = y$. **a** Plot of calibration data, simulating movement onset, constituted by the sEMG envelope e_t^{cal} with $I = 1$ channel (upper blue) and the concurrent movement instruction y_t^{cal} with $J = 1$DoF (lower red). **b** Plot of the learned mapping $\widehat{y} = f_1(e, \boldsymbol{\theta})$ performed by an MRL network calibrated by minimizing $\mathcal{L}_i$ with respect to $\boldsymbol{\theta}$. This model approximates a categorical decision threshold and does not enable proportionality. **c** Plot of the learned mapping $\widehat{y} = f_2(e, \boldsymbol{\theta})$ performed by an MRL network calibrated by instead minimizing $\mathcal{L}_i + \alpha_c \mathcal{L}_c$ with respect to $\boldsymbol{\theta}$. This model produces output which varies smoothly with latent muscle activity and thus enables proportionality

previously unobserved contraction intensities, the total loss function contains a second, regularizing term, referred to as the contractive loss, denoted by $\mathcal{L}_c$, and defined in (5).

$$\mathcal{L}_c(\boldsymbol{E}^{cal}, \boldsymbol{\theta}) = \frac{1}{TIJ} \sum_{t}^{T} \sum_{i}^{I} \sum_{j}^{J} \left(\frac{\partial \widehat{y}_j}{\partial e_i}\bigg|_{e_i = \boldsymbol{E}_{t,i}^{cal}} \right)^2 \quad (5)$$

$\mathcal{L}_c$ here represents the squared gradient of every individual output element $\widehat{y}_j$ with regard to every individual input element e_i, computed at the value generated from the presented inputs (i.e. the columns of $\boldsymbol{E}^{cal}$) and averaged across all such provided calibration examples. Stated equivalently, $\mathcal{L}_c$ is Frobenius norm of the Jacobian matrix of the multivariate function performed by (the current parameter configuration of) the network [18], averaged across all calibration example instances $\{\boldsymbol{e}_t^{cal}, \boldsymbol{y}_t^{cal}\}, t \in [1, T]$.

To understand the consequences of a minimized value of $\mathcal{L}_c$, it is important to note that the mapping performed by any ANN model with fully differentiable activation functions (i.e. compatible with backpropagation) is, by definition, continuous in the mathematical sense [18]. However, without any constraints, such a model may still learn parameter values $\boldsymbol{\theta}$ such that the gradient $\partial \widehat{y}_j / \partial e_i$ of any element $\widehat{y}_j$ in the output with regard to any element e_i in the input becomes arbitrarily large at any number of points in the space of possible inputs. To counteract this sometimes unwanted property by 'incentivizing' models to learn to associate limited deviations in input with only limited deviation in output, contractive loss of the same formulation as that of (5) has previously been applied as a form of regularization for the class of unsupervised neural network models known as *contractive autoencoders* [50].

The postulated relevance of $\mathcal{L}_c$ in the current problem formulation lies in the common assumption that muscle kinetics, and by extension limb kinematics, correlate

(nonlinearly) with the concurrent signal envelope as extracted from sEMG collected at relevant recording sites [40]. Formulated in the MRL framework, this assumption can be stated differently: increases or decreases of the values constituting $\hat{\boldsymbol{y}}$ should only occur in conjunction with, and (approximately) monotonically with, increases and decreases, respectively, of the values constituting $\boldsymbol{e}$. A network with this property will both give rise to proportionality of control output and achieve a small value for $\mathcal{L}_c$, thereby warranting the inclusion of this loss term here. A specific example illustrating how minimizing $\mathcal{L}_c$ induces proportionality can be viewed in Fig. 2.

With the total loss $\mathcal{L}$ established as above, its minimization with respect to $\boldsymbol{\theta}$ is performed iteratively via error backpropagation and the AdamW algorithm [55] (requiring hyperparameters η, β_1, β_2, and weight decay λ) for gradient descent in minibatches of size B (a hyperparameter). The minibatch affiliation of each individual calibration example $\{\boldsymbol{e}_t^{cal}, \boldsymbol{y}_t^{cal}\}$ is determined randomly at the start of every training epoch. All network weights $\boldsymbol{W}$ are initialized randomly via Glorot initialization [56] and all biases $\boldsymbol{b}$ are initially set to 0.

Prior to being fed into the ANN model for loss evaluation and parameter updating, the input sEMG envelope time slices of each minibatch is at every iteration corrupted additively with isotropic white Gaussian noise of predefined variance σ^2 (a hyperparameter). This step, inspired by its analogue in denoising autoencoders [57], has a twofold purpose: Firstly, it acts as a form of data augmentation, whereby the model overfitting to spurious patterns in the calibration data is made more unlikely [18]. Secondly, an ANN model calibrated with such input data will learn to map elements close to each other in the space of possible inputs $\boldsymbol{e}$ to elements close to each other in the space of possible outputs $\boldsymbol{y}$ [57]. In conjunction with the minimization of the contractive loss $\mathcal{L}_c$, this effect contributes to calibrated models which has the here desired property of proportionality.

Before the calibration process begins, a percentage P (a hyperparameter) of the available calibration data $\{\boldsymbol{E}^{cal}, \boldsymbol{Y}^{cal}\}$ is sampled randomly, without replacement, to be held out and used as validation data $\{\boldsymbol{E}^{val}, \boldsymbol{Y}^{val}\}$. At the conclusion of every Adam update iteration, this validation data is used to compute a validation error $\mathcal{L}_v = \mathcal{L}(\boldsymbol{E}^{val}, \boldsymbol{Y}^{val}, \boldsymbol{\theta})$ of the model parameter configuration $\boldsymbol{\theta}$ at that iteration. The optimization process continues until $\mathcal{L}_v$ is greater than that obtained V iterations previously (i.e. a form of early stopping) or until a total of M parameter update iterations have been performed, whichever comes first (M and V are hyperparameters). At this point, the model is considered calibrated and can be used for real-time inference of continuously encoded movement intent (i.e. kinematics).

Benchmark myoelectric pattern recognition framework

For the purpose of verifying the conjectured advantages of using MRL for myoelectric control, a proportional myocontrol pattern recognition method based on LDA is here chosen as the object of comparison due to its paradigmatic role in contemporary and commercially available prosthetic systems [41]. Implementation details and hyperparameter values are selected to be identical to those of Method 2 introduced by Scheme et al. in [40], as this approach, like MRL, only requires sEMG collected at a single level of muscle contraction for calibration. Furthermore, the selected method has been used in clinical settings for some time and thus represents a reliable application of pattern recognition for myoelectric control. For brevity, the method for motion-normalized proportional control denoted Method 2 in [40] will in its entirety henceforth be referred to simply as LDA. Importantly, LDA does not require any manual tuning of parameters (e.g. gains and thresholds), thus eliminating the risk that the experiment supervisor impacts the quality of the calibration results in the current study. A brief summary of this method is provided below.

In contrast to MRL, LDA does not operate on sEMG envelopes alone; instead, a sliding window is used to extract 4 time-domain features per channel from the raw sEMG signals: Mean Absolute Value (MAV), Zero Crossings, Slope Sign Changes, and Waveform Length, all introduced by Hudgins et al. in [12]. As LDA is not inherently simultaneous over the available DoFs, each detectable movement combination is instead assigned a unique categorical value to be inferred in order to allow for multiarticulate control. For a problem formulation involving J independently controllable, bidirectional DoFs, LDA thus operates to map each processed feature time window to a member of the set of 3^J possible movement classes (each bidirectional and categorical DoF can independently assume 3 mutually exclusive states). LDA achieves proportionality by computing a proportionality scalar PC_m, contingent on the index $m \in [0, 3^J - 1]$ of the currently inferred movement class, to each feature window. PC_m is computed in parallel with the running classification process by use of the MAV feature as detailed in (6), (7), and (8).

$$\mathrm{PC}_m = \left(\frac{1}{C_m} \sum_{i=1}^{I} S_{i,m} \mathrm{MAV}_i \right)^2 \tag{6}$$

$$C_m = \sum_{i=1}^{I} S_{i,m} \tag{7}$$

$$S_{i,m} = \frac{1}{K_m} \sum_{k=1}^{K_m} \mathrm{MAV}_{i,m,k}^{cal} \tag{8}$$

MAV_i is the MAV feature value of the i th sEMG channel of the processed window,$\mathrm{MAV}_{i,m,k}^{cal}$ is the MAV feature value of the i th sEMG channel of the k th feature window of the m th movement of the calibration data, and K_m is the total number of calibration data feature windows for movement m. As before, I is the number of sEMG channels. The training of the classification algorithm, together with the computation of the class centres $\boldsymbol{C} = [C_1 \cdots C_m \cdots C_M]$ and of the matrix $\boldsymbol{S} = \{S_{i,m}\}$, constitute the entirety of the LDA calibration which is performed on a subject-wise basis.

Experiments

Subjects

20 able-bodied subjects (age range 24–58 years, median age 32 years, 13 male, 7 female, 18 right-handed, 2 left-handed) without history of known neuromuscular or musculoskeletal disorders participated in the current study. The study was approved by the Regional Ethics Review Board in Lund, Sweden and was conducted in accordance with the tenets of the Declaration of Helsinki. All subjects were informed about the contents of the experiments, both verbally and in writing, and gave their informed and written consent.

Each subject participated in two separate experiment sessions: the first session, hereinafter referred to as day 1, consisted of both calibration data acquisition, model calibration, and evaluation of the two methods (MRL and LDA) for myoelectric control. This first session lasted for a total duration of 1 h or less, with some variations across subjects. The second session, carried out one week later and hereinafter referred to as day 7, entailed evaluation of both methods without any recalibration (in order to quantify interface stability) and thus lasted for a shorter duration (approximately 30 min). None of the subjects had prior experience with the studied myoelectric control methods.

Calibration data acquisition

A Myo armband (Thalmic Labs, Canada), composed of 8 circularly arranged dry surface electrodes of size 100 mm^2, was used to acquire sEMG time series from

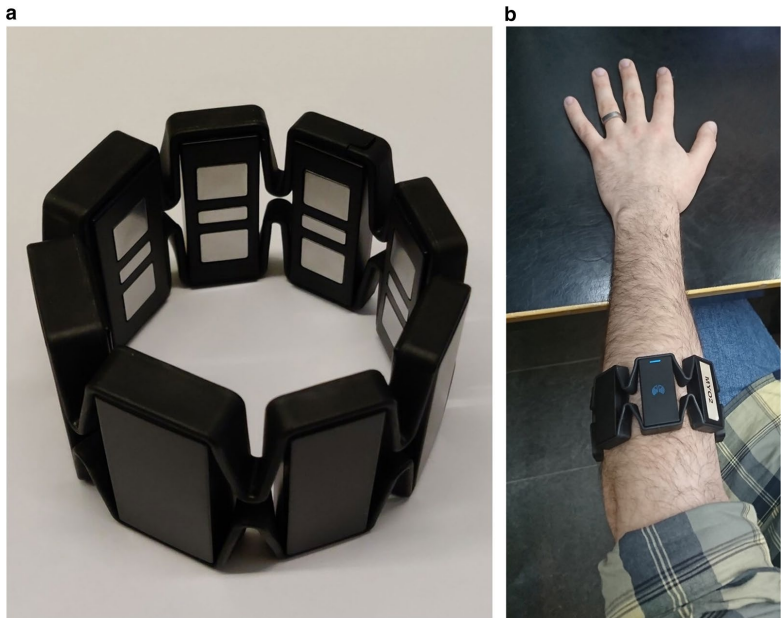


Fig. 3 The acquisition setup. Photographs depict **a** the Myo armband used for recording sEMG signals and **b** the placement of the Myo armband on the arm of a subject

subjects (see Fig. 3a). Prior to A/D conversion, EMG voltage signals were filtered using a built-in analogue bandpass filter with passband 5–100 Hz and a 50 Hz notch filter. Digital signals were sampled with 8-bit precision at a rate of $F_s = 200$ Hz and transferred at that same rate to a host desktop computer (on which all signal processing was performed) wirelessly via Bluetooth. The Myo armband was placed enclosing the dominant forearm of the subject at a level approximately 1/3 of the distance from the elbow joint to the wrist joint (see Fig. 3b). A photograph depicting the armband position and orientation was taken on day 1 in order to provide guidance for the redonning on day 7. The process of donning the armband could always be completed in a time of at most one minute. Following the placement of the armband, the subject was seated comfortably at approximately 1 m distance from a computer screen with the elbow resting on a table placed in front of the subject. The subjects could freely vary the position and angle of the elbow joint during all parts of the experiment.

The current study concerned the independent control of 2 separate DoFs: flexion and extension of the wrist (communicated to subjects as 'wrist right' and 'wrist left', contingent on the handedness of the subject), and flexion and extension of all digits simultaneously (communicated to subjects as 'hand close' and 'hand open', respectively). A ternary movement encoding approach was employed in the current study, resulting in $3^2 = 9$ possible compound movements (listed in Table 1).

Prior to calibration data acquisitions, sEMG was recorded while the subject performed all movements, excluding rest, with maximum voluntary contraction for 5 s. This step served to familiarize the subject with the movement combinations under consideration and was used to compute a maximum voluntary contraction (MVC) signal magnitude value specific to each movement by summing the MAV over all 8 sEMG channels.

Calibration data was recorded via an automated acquisition program which prompted the subject to perform all nonrest movements for 3 repetitions, each lasting for a duration of 5 s and separated by 3 s of rest. To aid the subject in applying a sustainable and consistent level of contraction across movements, the MAV of the EMG signal, extracted via a sliding window of length 0.5 s and summed over all channels, was mapped to the height of a bar shown in real-time on the computer screen together with a threshold set to equal 50% of the movement-specific MVC magnitude computed earlier; subjects were instructed to keep the height of the bar as close to the threshold as possible. Once the program was concluded, recorded sEMG was saved together with synchronized movement instruction stimuli signals. An example of such calibration data from a single subject is shown in Fig. 4.

Table 1 The calibration movements for right-handed subjects, all recorded on day 1

Movement class	Description	Ternary movement encoding
$m = 0$	Rest	$\boldsymbol{y} = [0, 0]$
$m = 1$	Wrist flexion	$\boldsymbol{y} = [-1, 0]$
$m = 2$	Wrist extension	$\boldsymbol{y} = [1, 0]$
$m = 3$	Flexion of the digits	$\boldsymbol{y} = [0, -1]$
$m = 4$	Extension of the digits	$\boldsymbol{y} = [0, 1]$
$m = 5$	Wrist flexion and Flexion of the digits	$\boldsymbol{y} = [-1, -1]$
$m = 6$	Wrist flexion and extension of the digits	$\boldsymbol{y} = [-1, 1]$
$m = 7$	Wrist extension and flexion of the digits	$\boldsymbol{y} = [1, -1]$
$m = 8$	Wrist extension and extension of the digits	$\boldsymbol{y} = [1, 1]$

Calibration of models

The MRL model and the LDA model described previously were both automatically calibrated immediately following the conclusion of the data acquisition phase. The methods were executed with Python 3.6, using the SKLearn library [58] to implement the LDA model and the TensorFlow [59] library to implement the MRL model. No part of the calibration of either method required any manual intervention by the experiment supervisor.

The hyperparameter of the MRL framework had been selected empirically based on data from pilot work with subjects who were not participating in the current study and were not subject to change for the duration of the experiments; exact numerical values are presented in Table 2. Analysis of the ANN topology resulting from this configuration using the computational cost estimation heuristics from [27] revealed an approximate inference-mode time complexity of 5.5 MFLOPS (at 200 forward passes per second, i.e. synchronously with sEMG sampling) and a memory footprint of 59.5 kB (excluding the computational cost of preprocessing); these are requirements compatible with contemporary embedded systems. Furthermore, the use of a preprocessing filter length of $W = 0.5$ s (100 samples at $F_s = 200$ Hz) corresponds to a group delay (i.e. lag) of $(100 - 1)/(2F_s) = 0.2475$ s for a FIR filter [60]—less than the typically desired maximum delay of 300 ms in myocontrol applications [61].

Calibration of the MRL model was performed according to the procedures outlined previously in the section titled 'Calibration'. The average (across subjects) wall time required to calibrate a single MRL model with data collected from a single subject on the desktop computer

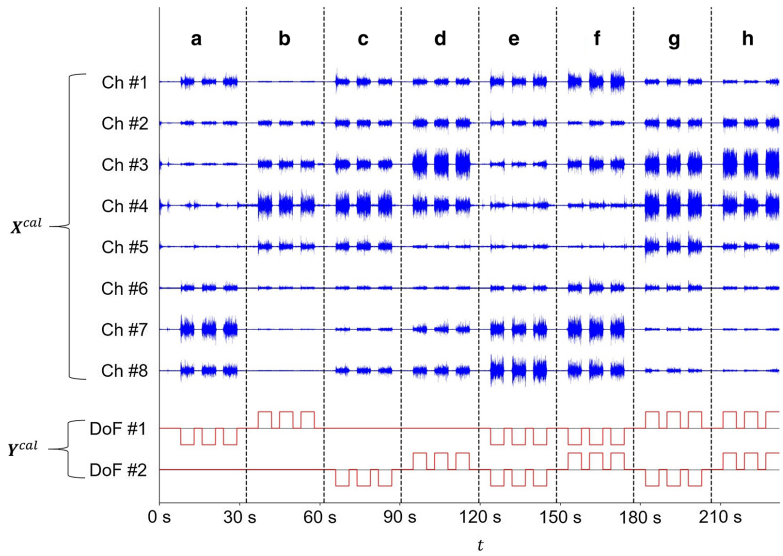


Fig. 4 Calibration signals acquired from a representative, right-handed subject. The calibration data is constituted by raw sEMG ($\boldsymbol{X}^{cal}$) and trinary (−1, 0, or 1) DoF- and sample-wise encodings of concurrent movement instruction stimuli ($\boldsymbol{Y}^{cal}$). **a** Wrist flexion. **b** Wrist extension. **c** Flexion of the digits. **d** Extension of the digits **e** Wrist flexion and flexion of the digits. **f** Wrist flexion and extension of the digits. **g** Wrist extension and flexion of the digits. **h** Wrist extension and extension of the digits

Table 2 The hyperparameters of the MRL framework and their respective values selected for the current study

Hyperparameter	Symbol	Value
Floating point precision	b	32 bits
Number of sEMG channels	I	8
Number of decodable DoFs	J	2
Envelope extraction filter length	W	0.5 s
Size of first encoder layer	2^K	128
Encoder network depth	N	5
Code size	2^{K-N+1}	8
Decoders hidden layer size	2^S	32
Contractive loss weigh	α_c	10^{-2}
Adam hyperparameters	η	10^{-4}
	β_1	0.9
	β_2	0.999
Corruptive noise variance	σ^2	10^{-1}
Weight decay	λ	10^{-6}
Minibatch size	B	2^{12}
Validation set percentage	P	10%
Validation lookback	V	300
Maximum number of iterations	M	5000

(equipped with a Nvidia Titan V GPU) was measured as 97.96 s (SD 0.43 s). An illustration of the mapping performed by a calibrated MRL model is presented in Fig. 5.

LDA was calibrated as per the specifications of the previously summarized original study [40], with hyperparameter values as those of the same study, using a feature window of duration 160 ms (32 samples) and inter-window time increments of 15 ms (3 samples). The feature window was moved across the entirety of each collected EMG without regard to when individual movement repetitions started and ended. All $8 \cdot 4 = 32$ features were individually renormalized to have zero mean and unit variance across all calibration data feature window locations. During real-time inference, features were similarly rescaled with the calibration set means and variances prior to being processed by the classifier. The ground truth movement class m of each calibration data feature window was determined via a majority vote over its constituent time samples, thereby resolving ambiguities regarding the class affiliation of feature windows containing a shift in contraction level. The average (across subjects) wall time required to calibrate the LDA model to a single subject was measured as 1.31 s (SD 0.09 s).

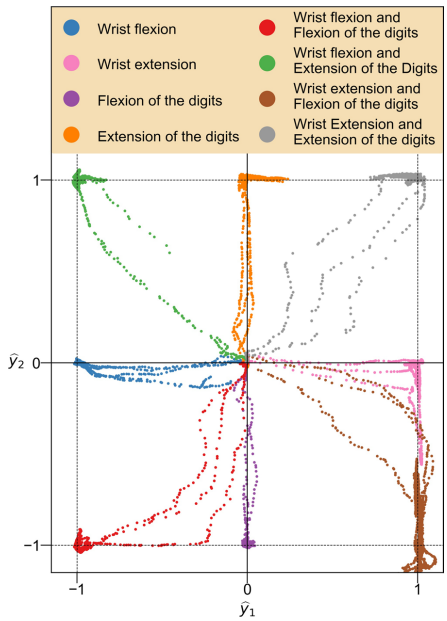


Fig. 5 Example of MRL calibration results from a single subject. Each point represents the output $\hat{\mathbf{y}} = [\hat{y}_1, \hat{y}_2]^T$ of the calibrated MRL model when fed an 8-channel sEMG sample from the calibration data, coloured according to the movement instruction stimuli with which it cooccurred. Each 'trail' connecting the origin to a cluster represents the onset and conclusion of a movement repetition

Although previous studies (e.g. [33]) have made use of the same combination of feature set and acquisitions setup as the LDA benchmark framework of the current study, concerns can be raised over the appropriateness of the selected time-domain features due to the relatively limited sampling rate of the system (200 Hz). This, in turn, could potentially make comparisons between MRL and LDA unbalanced, as the former utilizes high frequency information to a lesser degree than the latter. To verify the absence of any such adverse effects on the LDA classifier originating from insufficient signal bandwidth, 10% of calibration examples (selected randomly) were for each subject withheld during calibration and used to compute a validation accuracy. The average (across subjects) validation accuracy obtained in this way was 92.71% (SD 4.05%). This level of performance is in line with what is expected in light of previous studies with comparable number of detectable movement classes [10], indicating that the acquisition setup provided sufficient information for the selected feature set and classifier.

Evaluation

Common offline measures of inferential performance (e.g. classification accuracy) have often been found to exhibit only limited correlation with the quality of myoelectric control as estimated by functional tests [10]. Efficacy of control was for this reason here instead evaluated using a type of real-time test originally introduced by Williams and Kirsch in [62] and based on Fitts's law [48]. The test makes use of a human test subject to leverage the myoelectric control method under investigation to steer a cursor towards a set of circular targets presented to the subject on a computer screen. The subject and the control method are thereafter evaluated in unison via the calculation of a set of performance metrics; each metric aggregated over multiple subjects is assumed to constitute an informative measure of control method quality. This particular approach to the evaluation of man–machine control interfaces has been validated for use with control based on forearm myoelectricity multiple times (e.g. [38, 44]).

In the current study, the performance of each subjects with both myoelectric control methods were evaluated in sequence. To eliminate confounding effects on measured performance originating from user adaptation, 10 subjects were randomly selected to be evaluated using MRL first and then using LDA, whereas the remaining 10 subjects were selected to be evaluated using LDA first and then using MRL. All subjects were blind to the order in which the methods were presented to them. On day 1, the evaluation phase was undertaken directly following data acquisition and model calibration, with the Myo armband unmoved. On day 7, the evaluation phase followed directly after the initial redonning of the Myo armband. Evaluative tests were otherwise conducted identically on day 1 and on day 7.

The control interface

For both LDA and MRL, detection of wrist flexion and wrist extension corresponded to cursor translations left and right, respectively, whereas detection of flexion of the digits and extension of the digits corresponded to cursor translations down and up, respectively (as shown in Fig. 6). For LDA, the direction of cursor translation was determined according to the detected movement class, with 1-DoF movements leading to cursor translation parallel to the screen coordinate axes and 2-DoF movements leading to cursor translations parallel to the axes' diagonals. Cursor speed was scaled linearly such that $PC_m = 0$ (as defined in (6)) resulted in a speed of 0 pixels per second and $PC_m = 1$ resulted in a speed of 540 pixels per second. Detection of the rest class resulted in a cursor speed of 0 pixels per second. For MRL, the velocities of the cursor in the x- and y directions were separately set

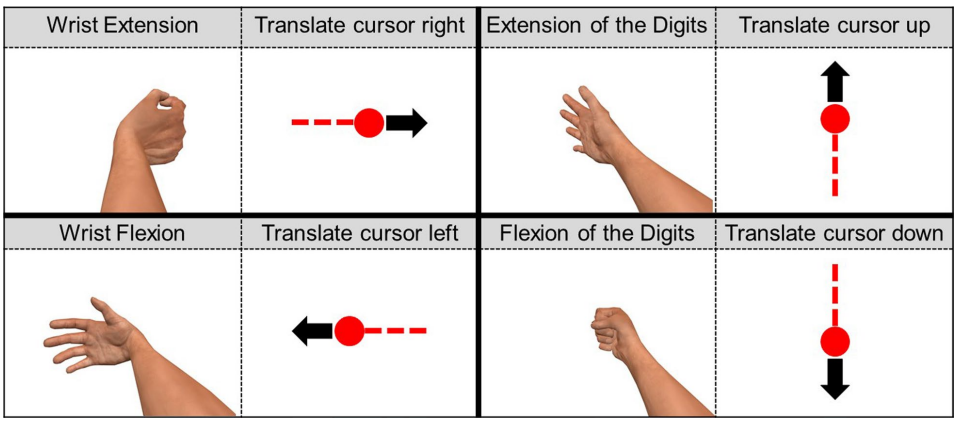


Fig. 6 Decodable DoFs and corresponding cursor translations in the test environment for a right-handed subject. Gestures incorporating combinations of the DoFs allowed for cursor translations in intermediate directions

to equal the first and second element of the ANN-estimated kinematics $\hat{\boldsymbol{y}}$, respectively. The cursor speed was scaled linearly such that $\|\hat{\boldsymbol{y}}\|_2 = 0$ resulted in a speed of 0 pixels per second and $\|\hat{\boldsymbol{y}}\|_2 = 1$ resulted in a speed of 540 pixels per second. Although MRL in principle allows for position control, whereby $\hat{\boldsymbol{y}}$ are mapped directly to the x- and y-coordinates of the cursor, the decision was made to consistently use this type of velocity control in order to be able to fairly compare results with those obtained with LDA.

Acclimation

Prior to evaluation, subjects underwent a brief acclimation exercise to become familiarized with the control interface. This preparatory step was intended to mitigate the impact of user adaptation during subsequent tests. The exercise consisted of steering the cursor towards 8 targets presented to the subject one at a time; 4 targets were placed along the x- and y axes with peripheries touching the screen edge, and 4 targets were placed similarly at the ends of the coordinate axes' diagonals. Once each target had been reached 5 times the exercise was considered complete. The acclimation exercise, which lasted for 3–5 min, was performed prior to the Fitts's law test on day 1 and day 7 for both MRL and LDA.

The performance test

Every combination of 20 possible center coordinates (5 per coordinate system quadrant) and 2 possible radii were selected as targets for the Fitts's law test, resulting in $20 \cdot 2 = 40$ unique targets (all shown in Fig. 7). The possible radii were selected as 60 pixels and $\lceil \sqrt{2} \cdot 60 \rceil = 85$ pixels on a monitor with resolution 1920×1080 pixels, corresponding to circular targets covering 0.6% and 1.2% of the total screen area visible to the subject, respectively.

In an order established randomly for each subject, targets were one by one plotted on the screen; the cursor was at this time held stationary at the origin. After 3 s (selected to minimize the impact of reaction time and planning), a vibration cue was given to the user through the Myo armband, and the subject was granted control of the cursor. The instruction of the subject was at this time to move the cursor to the presented target as rapidly as possible. As in previous studies [38, 44], the target was considered successfully reached once the cursor had resided within its boundary for a dwell time of 0.3 s. If 20 s passed without the subject successfully reaching the target, the task was reported as failed. Following success or failure, the target was removed from the screen, the cursor recentred, and the subject prompted to rest for 5 s, after which a new target was presented. The procedure was repeated until all 40 targets-reaching tasks had been attempted exactly once by the subject.

When all targets had either been successfully reached or failed, a set of 5 performance metrics, named completion rate (CR), completion time (CT), path efficiency (PE), overshoot (O), and throughput (T), was calculated from the cursor trajectories traversed during the test. These metrics, defined in [62] and summarized in Table 3, characterize separate aspects of the performance of the test subject and control method across all targets.

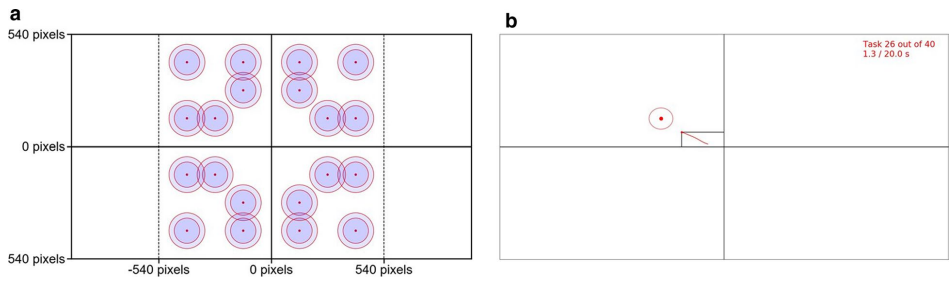


Fig. 7 The Fitts's law test. **a** The 40 combinations of target positions and radii selected for the current study. No target was placed on any of the coordinate axes, thus requiring the subject to activate both DoFs, either sequentially or simultaneously, for successful task completion. **b** Screen capture of the view of a subject while steering the cursor towards one of the targets presented during the test. The red 'tail' tracking the cursor represents the positions occupied during the preceding 0.3 s (60 samples at the 200 Hz sampling rate). A target was considered reached after a dwell time of 0.3 s, i.e. when the entirety of the tail was situated within the target periphery

Table 3 The real-time myocontrol performance metrics calculated for each subject and control method

Metric	Abbreviation	Summary
Completion rate [%]	*CR*	The proportion of targets which are successfully reached
Completion time [s]	*CT*	The total time from the start of the task to the target being reached, averaged across all targets (excluding failed attempts)
Path Efficiency [%]	*PE*	The ratio between the length of the optimal (straight line) path from the origin to the target and the distance traversed by the cursor, averaged across all targets
Overshoot	*O*	Total number of occurrences during the test wherein the cursor leaves the target prior to dwell time elapsion, divided by the total number of targets
Throughput [bits/s]	*T*	The ratio between the (target-wise) index of difficulty and the (target-wise) completion time averaged across all successfully reached targets

To calculate T, the target-specific index of difficulty (ID) was defined as in [62] for each of the 40 targets.

$$\mathrm{ID} = \log_2\left(\frac{\mathrm{D}}{\mathrm{W}} + 1\right) \tag{9}$$

D is the Euclidean (straight-line) distance from the origin to the target and W is the diameter of the target. Each of the 40 targets were thus assigned an ID out of $3 \cdot 2 = 6$ possible values (from the 3 values of D and the 2 values of W visible in Fig. 7) ranging from 1.17 bits to 2.38 bits.

The gold standard

Once both myoelectric control methods had been evaluated on day 1, each subject was instructed to complete the aforementioned performance test using a regular computer mouse. As the computer mouse did not allow for vibration stimuli, task onset was instead accompanied by a visual trigger presented on the screen. Just as for myocontrol evaluation, targets were considered reached after a dwell time of 0.3 s (i.e. no clicking was required) and considered failed after 20 s. This step was undertaken once per test subject and was not repeated on day 7. The same set of 5 performance metrics was calculated from the obtained trajectories with the purpose of contrasting myocontrol efficacy with that of the arguably best available tool of contemporary man–machine interfacing – the performance achieved with a computer mouse during this step is for this reason denoted the gold standard (GS).

Statistics

The analysis described in this section was performed using SPSS Statistics version 27.0. Linear regression models with ID as independent variable and target-wise completion time as dependent variable was fitted for MRL and LDA independently using all targets from all session and subjects. As in previous studies [38, 44], a high value of R^2 was seen as indicative of the validity of the Fitts's law test.

Table 4 Means and standard deviations of performance metrics obtained on day 1

Metric	Method		
	LDA	MRL	GS
CR[%]	97.00 ± 4.78	99.38 ± 1.34	100.0 ± 0.0
CT[s]	5.23 ± 1.67	3.73 ± 1.51	1.23 ± 0.16
PE[%]	41.42 ± 8.58	49.20 ± 10.91	82.17 ± 1.96
O	0.60 ± 0.32	0.36 ± 0.24	0.00 ± 0.00
T[bits/s]	0.51 ± 0.12	0.69 ± 0.22	1.62 ± 0.20

Table 5 Means and standard deviations of performance metrics obtained on day 7

Metric	Method	
	LDA	MRL
CR[%]	97.38 ± 5.27	99.63 ± 1.19
CT[s]	5.38 ± 1.79	3.45 ± 1.29
PE[%]	38.75 ± 9.15	50.73 ± 9.13
O	0.59 ± 0.26	0.42 ± 0.25
T[bits / s]	0.49 ± 0.13	0.71 ± 0.20

Full factorial two-way repeated measures multivariate analysis of variance (MANOVA), with independent variables method (with two levels: LDA and MRL) and session (with two levels: day 1 and day 7) and all 5 performance metrics as dependent variables, was used to simultaneously assess the impact of control approach and time since calibration on aggregated control efficacy. p-values of less than 0.05 after Bonferroni correction for 3 estimated terms (method, session, and method*session) were considered significant. Due to the statistical significance of results obtained via MANOVA (see the Results section), this step was followed by post-hoc metric-wise univariate analysis of variance (ANOVA) in order to assess the impact of the independent variables on each metric separately. As with MANOVA, two-way full factorial repeated measure designs with a $p < 0.05$ significance level (subject to Bonferroni correction for multiple comparisons [63]) were employed for this step. In addition to the p-values obtained via MANOVA and ANOVA, the Cohen's d effect size [64] separating MRL from LDA was computed for each metric using the full concatenation of results obtained on day 1 and on day 7.

Results

For MRL, a strong linear relationship (coefficient of determination $R^2 = 0.97$ with $p = 0.0010$) was observed between ID and CT. The corresponding value for LDA

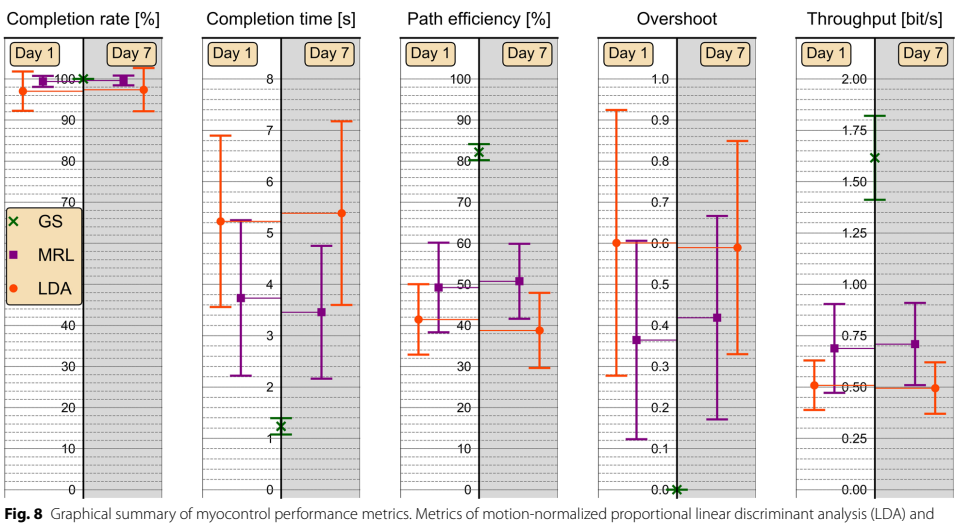


Fig. 8 Graphical summary of myocontrol performance metrics. Metrics of motion-normalized proportional linear discriminant analysis (LDA) and the proposed myoelectric representation learning (MRL) method were obtained from evaluations conducted immediately following calibration (day 1) and after one week of intermission (day 7). Performances achieved by test subjects using the gold standard (GS), i.e. a computer mouse, are included for reference. Markers and error bars represent arithmetic means and standard deviations, respectively, across all 20 test subjects

was computed as $R^2 = 0.89$ with $p = 0.0019$, lending credence to a view of a Fitts's law test as suitable for method evaluation and by extension the appropriateness of throughput T as a measure of overall performance.

Summary statistics of performance metrics obtained with all methods during day 1 and during day 7 are presented in Table 4 and in Table 5, respectively; the same data are summarized graphically in Fig. 8. MANOVA detected a significant effect of method ($p = 0.012$) on the full set of 5 performance metrics. No significant effect of neither session ($p = 1.00$) nor the interaction term method*session ($p = 0.070$) was detected.

For *CR*, ANOVA detected a significant effect of method ($p = 0.044$) and non-significant effects of session ($p = 1.00$) and method*session ($p = 1.00$). The effect size separating the mean value of CR across both sessions for MRL (99.50%) from that of LDA (97.19%) was computed as $d = 0.62$.

For *CT*, ANOVA detected a significant effect of method ($p = 0.000042$) and non-significant effects of session ($p = 1.00$) and method*session ($p = 0.62$). The effect size separating the mean value of *CT* across both sessions for MRL (3.59 s) from that of LDA (5.31 s) was computed as $d = -1.07$.

For *PE*, ANOVA detected a significant effect of method ($p = 0.00030$) and non-significant effects of session ($p = 1.00$) and method*session ($p = 0.27$). The effect size separating the mean value of *PE* across both sessions for MRL (49.96%) from that of LDA (40.09%) was computed as $d = 1.02$.

For *O*, ANOVA detected a significant effect of method ($p = 0.016$) and non-significant effects of session ($p = 1.00$) and method*session ($p = 1.00$). The effect size separating the mean value of *O* across both sessions for MRL (0.39) from that of LDA (0.60) was computed as $d = -0.74$.

For *T*, ANOVA detected a significant effect of method ($p = 0.00033$) and non-significant effects of session ($p = 1.00$) and method*session ($p = 0.97$). The effect size separating the mean value of *T* across both sessions for MRL (0.69 bits/s) from that of LDA (0.50 bits/s) was computed as $d = 1.13$.

Discussion

The main aim of the current study was to empirically evaluate the hypothesized advantages of using the proposed MRL framework for myoelectric control. For all considered performance metrics, a significant advantage was detected for MRL over LDA. For the *T* metric, which characterize overall efficacy of control in terms of transmitted information, the effect size separating MRL from the benchmark LDA method was computed as $d = 1.13$, showing that subjects achieved slightly more than one standard deviation higher performance with MRL than with LDA. For metrics characterizing more specific aspects of myoelectric control quality, the results show that the performance of MRL surpassed that of LDA, although sometimes to a lesser (but still consistently significant) degree. Anecdotally, all but 2 subjects independently expressed preference for MRL over LDA while still blind to the experimental condition, some noting that control with the latter sometimes resulted in unpredictable cursor movements, whereas the former allowed for 'smoother' steering and was conducive to 'course-corrections' if the cursor did not follow the initially planned trajectory.

Contrary to what was expected in light of some previous findings [33], the current study failed to detect any statistically significant deterioration in performance over time for either control method (neither MANOVA nor ANOVA detected any significant effect of session or method*session on any metric). A potential explanation for this discrepancy lies in the properties of the EMG recording setup used in the current study: as the electrodes had relatively large pickup area compared to what is typical, the signals they acquired could plausibly be resistant to small variations in electrode positioning over time [65]. Although it is also consistent with previous findings [66–68] to assume that subjects underwent continuous motor learning, which potentially obscures the effects of drift in EMG distribution, the results are nevertheless encouraging in that they indicate that MRL does not require frequent recalibration in order to retain its advantages over the LDA approach.

In addition to validating the MRL framework, the findings presented in the current study support a favourable view of myoelectric control based on regression of kinematics (i.e. proportional estimation of multiple DoFs simultaneously) more generally. As was experienced by the subjects of the current study, incorrect estimations can have a substantial effect on the perceived quality of control when the space of possible movements is quantized, as is the case with LDA. Conversely, errors on the part of the continuous MRL algorithm were perceived as easier to counteract, allegedly due to the less 'jittery' nature of the resulting interface. Compared to many conventional methods with this advantage, the type of regression proposed in the current study requires neither continuous target values nor mirrored training, allowing for straightforward use by both unilateral and bilateral amputees.

In a manner similar to the current study, one previous approach has applied regression models calibrated with entirely categorical target values for the task of myoelectric control of hand prostheses (see e.g. [69]). This earlier approach, termed on–off goal-directed training

[70], applies ridge regression to estimate a linear mapping from EMG features to concurrent multi-label visual movement instruction stimulus. Once calibrated, the linear regression model use EMG to infer continuous force-patterns exerted by the digits in real-time. Notably, due to its assumptions, this method requires an approximately linear relationship between the selected EMG features and activation intensity to hold true for all steerable output DoFs in order to function effectively. This is not the case for the MRL framework introduced in the current study: a large class of possible nonlinear relationships between EMG and kinematics can be learned by the proposed ANN model during calibration. Consequently, the nonlinearity of the relationship between EMG and kinematics need not be fully captured by the selection of EMG features (i.e. the signal envelope in the current study) used as input to the MRL network model.

A salient limitation of the MRL framework as presented here relates to scalability with regard to the number of decodable DoFs. The current study was successful in extracting kinematics concerning two DoFs, but required calibration data of every possible movement combination, resulting in the recording of $3^2 = 9$ movements. As the number of performable movement combinations 3^J grows geometrically with the number of independent DoFs J, larger numbers of DoFs quickly lead to infeasible calibration data acquisition durations. This drawback is not unique to MRL but is, to the best of our knowledge, shared by all contemporary pattern recognition frameworks aimed at multiarticulate control. Future work could potentially focus on solving this problem by formulating generative sEMG models to artificially provide compound movement calibration data using subject-specific signals acquired exclusively from 1-DoF movements.

One avenue of algorithmic improvement for MRL concerns the preprocessing of sEMG signals. In the current study, the choice was made to let the ANN model operate on sEMG envelopes; this was motivated by the observed monotonic relationship between kinematics and sEMG magnitude, which is a necessary property to allow for the application of proportionality-inducing contractive regularization. However, a body of previous work (cf. [11]) has been unanimous in establishing that higher frequency content of sEMG signals reflects factors of the movement-dependent generative process, making information encoded at such frequencies discriminative for the purpose of movement decoding. With the MRL approach introduced and used in the current study, most high frequency information is discarded (due to envelope extraction) and cannot impact the estimation of kinematics. A question which could warrant further investigation is whether MRL can be extended to include more sophisticated sEMG signal features (handcrafted or learned) while at the same time guaranteeing proportionality by keeping the regression output constrained by the magnitude of the sEMG envelope.

Conclusions

This paper has introduced an algorithm (MRL), based on regularized multitask learning, for the purpose of extracting proportionally encoded hand- and wrist movement intent pertaining to multiple DoFs from the forearm sEMG. To investigate the suitability of the proposed framework for use with muscle-computer interfaces, it was evaluated on able-bodied subjects with a Fitts's law type test and compared to a standard approach, based on LDA, for myocontrol. MRL was found to be superior to LDA in the sense of significantly outperforming the latter in all considered metrics of real-time efficacy of control. Furthermore, neither MRL nor LDA could be demonstrated to undergo significant deterioration in any performance metric over a time of 7 days. Although these findings are promising, future work will have to examine performance over a longer time period in order to ascertain long-term stability.

In addition to its advantages related to efficacy of control, the proposed MRL system is computationally lightweight and can operate in real-time with relatively restricted hardware resources. Future work could thus focus on implementing this approach in the context of wearable computing platforms without expectations of reduced quality of control. Such endeavours should ideally study control of physically instantiated robotic limbs and involve forearm amputees.

Abbreviations

ANN: Artificial neural network; ANOVA: Analysis of variance; CR: Completion rate; CT: Completion time; DoF: Degree of freedom; EMG: Electromyogram; GS: Gold standard; ID: Index of difficulty; LDA: Linear discriminant analysis; MANOVA: Multivariate analysis of variance; MAV: Mean absolute value; MCI: Muscle–computer interface; MRL: Myoelectric representation learning; MVC: Maximum voluntary contraction; O: Overshoot; PE: Path efficiency; T: Throughput.

Acknowledgements

The authors gratefully acknowledge the support of the NVIDIA Corporation with the donation of the Titan V GPU used for this research.

Authors' contributions

All authors contributed substantially to the design of the study. AO conceived, designed, and implemented the biosignal acquisition, data processing, and machine learning algorithms. AO, NM, CA ran the experiments. AO analysed the results, created the figures, and drafted the manuscript. NM, AB, CA provided critical comments and finalized the manuscript for publication. All authors read and approved the manuscript.

Funding

Open access funding provided by Lund University. This work was supported financially by the Promobilia Foundation, the Crafoord Foundation,

the European Commission under the DeTOP project (LEIT-ICT-24-2015, GA #687905), and the Swedish Research Council (DNR 2019-05601).

Availability of data and materials
All data and code used for this study are available from the corresponding author on reasonable request.

Ethics approval and consent to participate
This study was approved by the Regional Ethics Review Board in Lund, Sweden and was conducted in accordance with the tenets of the Declaration of Helsinki. All subjects were informed about the contents of the experiments, both verbally and in writing, and gave their informed and written consent.

Consent for publication
The manuscript does not contain any individual person's data.

Competing interests
The authors declare that they have no competing interests.

Author details
[1] Department of Biomedical Engineering, Faculty of Engineering, Lund University, Lund, Sweden. [2] Department of Hand Surgery, Institute of Clinical Sciences, Sahlgrenska Academy, Sahlgrenska University Hospital and University of Gothenburg, Gothenburg, Sweden. [3] Wallenberg Center for Molecular Medicine, Lund University, Lund, Sweden.

Received: 3 September 2020 Accepted: 2 February 2021
Published online: 15 February 2021

References

1. Kim DH, Lu N, Ma R, Kim YS, Kim RH, Wang S, et al. Epidermal electronics. Science. 2011;333:838–43.
2. Scott RN, Parker PA. Myoelectric prostheses: state of the art. J Med Eng Technol Informa Healthcare. 1988;12:143–51.
3. Paciga JE, Richard PD, Scott RN. Error rate in five-state myoelectric control systems. Biol Eng Comput. 1980;12:287–90.
4. Saikia A, Mazumdar S, Sahai N, Paul S, Bhatia D, Verma S, et al. Recent advancements in prosthetic hand technology. J Med Eng Technol. 2016;40:255–64.
5. Fougner A, Stavdahl O, Kyberd PJ, Losier YG, Parker PA. Control of upper limb prostheses: terminology and proportional myoelectric controla review. IEEE Trans Neural Syst Rehabil Eng. 2012;20:663–77.
6. Connolly C. Prosthetic hands from Touch Bionics. Ind Rob. 2008;35:290–3.
7. Biddiss E, Chau T. Upper-limb prosthetics: critical factors in device abandonment. Am J Phys Med Rehabil. 2007;86:977–87.
8. Farina D, Aszmann O. Bionic limbs: clinical reality and academic promises. Sci Transl Med. 2014;6:257–69.
9. Hakonen M, Piitulainen H, Visala A. Current state of digital signal processing in myoelectric interfaces and related applications. Biomed Signal Process Control. 2015;18:334–59.
10. Scheme E, Englehart K. Electromyogram pattern recognition for control of powered upper-limb prostheses: state of the art and challenges for clinical use. J Rehabil Res Dev. 2011;48:643–59.
11. Zecca M, Micera S, Carrozza MC, Dario P. Control of multifunctional prosthetic hands by processing the electromyographic signal. Crit Rev Biomed Eng. 2017;30:459–85.
12. Hudgins B, Parker P, Scott RN. A new strategy for multifunction myoelectric control. IEEE Trans Biomed Eng. 1993;40:82–94.
13. Khushaba RN, Al-Timemy AH, Al-Ani A, Al-Jumaily A. A framework of temporal-spatial descriptors-based feature extraction for improved myoelectric pattern recognition. IEEE Trans Neural Syst Rehabil Eng. 2017;25:1831–931.
14. Oskoei MA, Hu H. Support vector machine-based classification scheme for myoelectric control applied to upper limb. IEEE Trans Biomed Eng. 2008;55:1956–65.
15. Malešević N, Marković D, Kanitz G, Controzzi M, Cipriani C, Antfolk C. Vector autoregressive hierarchical hidden Markov models for extracting finger movements using multichannel surface EMG signals. Complexity. 2018. https://doi.org/10.1155/2018/9728264.
16. Shuman G, Durić Z, Barbará D, Lin J, Gerber LH. Improving the recognition of grips and movements of the hand using myoelectric signals. BMC Med Inform Decis Mak. 2016. https://doi.org/10.1186/s12911-016-0308-1.
17. Phinyomark A, Phukpattaranont P, Limsakul C. Feature reduction and selection for EMG signal classification. Expert Syst Appl. 2012;30:7420–31.
18. Ian G, Yoshua Bengio AC. Deep learning. Cambridge: MIT Press; 2015.
19. Geng W, Du Y, Jin W, Wei W, Hu Y, Li J. Gesture recognition by instantaneous surface EMG images. Sci Rep. 2016;6:36571.
20. Atzori M, Cognolato M, Müller H. Deep learning with convolutional neural networks applied to electromyography data: a resource for the classification of movements for prosthetic hands. Front Neurorobot. 2016;10:9.
21. Ameri A, Akhaee MA, Scheme E, Englehart K. Regression convolutional neural network for improved simultaneous EMG control. J Neural Eng. 2019;16:036015.
22. Olsson AE, Sager P, Andersson E, Björkman A, Malešević N, Antfolk C. Extraction of multi-labelled movement information from the Raw HD-sEMG image with time-domain depth. Sci Rep. 2019;9:7244.
23. Ameri A, Akhaee MA, Scheme E, Englehart K. Real-time, simultaneous myoelectric control using a convolutional neural network. PLoS ONE. 2018;13:e0203835.
24. Geng W, Hu Y, Wong Y, Wei W, Du Y, Kankanhalli M. A novel attention-based hybrid CNN-RNN architecture for sEMG-based gesture recognition. PLoS ONE. 2018;13:e0206049.
25. Olsson A, Malesevic N, Bjorkman A, Antfolk C. Exploiting the Intertemporal Structure of the Upper-Limb sEMG: comparisons between an LSTM Network and Cross-Sectional Myoelectric Pattern Recognition Methods. 41st Annual Int Conf IEEE Eng Med Biol Soc. 2019.
26. Phinyomark A, Scheme E. EMG pattern recognition in the era of big data and deep learning. Big Data Cogn Comput. 2018;2:21.
27. Olsson AE, Björkman A, Antfolk C. Automatic discovery of resource-restricted Convolutional Neural Network topologies for myoelectric pattern recognition. Comput Biol Med. 2020;120:103723.
28. Ortiz-Catalan M, Håkansson B, Brånemark R. Real-time and simultaneous control of artificial limbs based on pattern recognition algorithms. IEEE Trans Neural Syst Rehabil Eng. 2014;22:756–64.
29. Young AJ, Smith LH, Rouse EJ, Hargrove LJ. Classification of simultaneous movements using surface EMG pattern recognition. IEEE Trans Biomed Eng. 2013;60:1250–8.
30. Krasoulis A, Nazarpour K. Myoelectric digit action decoding with multi-label, multi-class classification: an offline analysis. bioRxiv. 2020;8:31.
31. Jiang N, Dosen S, Müller K-R, Farina D. Myoelectric control of artificial limbs—is there a need to change focus? IEEE Signal Process Mag. 2012;29:152–150.
32. Kyranou I, Vijayakumar S, Erden MS. Causes of performance degradation in non-invasive electromyographic pattern recognition in upper limb prostheses. Front Neurorobot. 2018;12:58.
33. Waris A, Mendez I, Englehart K, Jensen W, Kamavuako EN. On the robustness of real-time myoelectric control investigations: a multiday Fitts' law approach. J Neural Eng. 2019;16:026003.
34. Sensinger JW, Lock BA, Kuiken TA. Adaptive pattern recognition of myoelectric signals: exploration of conceptual framework and practical algorithms. IEEE Trans Neural Syst Rehabil Eng. 2009;17:270–8.
35. Hoozemans MJM, Van Dieën JH. Prediction of handgrip forces using surface EMG of forearm muscles. J Electromyogr Kinesiol. 2005;15:358–66.
36. Nielsen JLG, Holmgaard S, Jiang N, Englehart KB, Farina D, Parker PA. Simultaneous and proportional force estimation for multifunction myoelectric prostheses using mirrored bilateral training. IEEE Trans Biomed Eng. 2011;58:681–8.
37. Hahne JM, Markovic M, Farina D. User adaptation in myoelectric man–machine interfaces. Sci Rep. 2017;7:4437.
38. Jiang N, Rehbaum H, Vujaklija I, Graimann B, Farina D. Intuitive, online, simultaneous, and proportional myoelectric control over two degrees-of-freedom in upper limb amputees. IEEE Trans Neural Syst Rehabil Eng. 2014;22:501–10.
39. Kamavuako EN, Scheme EJ, Englehart KB. On the usability of intramuscular EMG for prosthetic control: a Fitts' Law approach. J Electromyogr Kinesiol. 2014;24:770–7.
40. Scheme E, Lock B, Hargrove L, Hill W, Kuruganti U, Englehart K. Motion normalized proportional control for improved pattern recognition-based myoelectric control. IEEE Trans Neural Syst Rehabil Eng. 2014;22:149–57.
41. Coapt Engineering. https://coaptgen2.com/. Accessed 7 Nov 2019.

42. Chen C, Chai G, Guo W, Sheng X, Farina D, Zhu X. Prediction of finger kinematics from discharge timings of motor units: implications for intuitive control of myoelectric prostheses. J Neural Eng. 2019;16:026005.
43. Smith LH, Kuiken TA, Hargrove LJ. Real-time simultaneous and proportional myoelectric control using intramuscular EMG. J Neural Eng. 2014;11:066013.
44. Vujaklija I, Shalchyan V, Kamavuako EN, Jiang N, Marateb HR, Farina D. Online mapping of EMG signals into kinematics by autoencoding. J Neuroeng Rehabil. 2018. https://doi.org/10.1186/s12984-018-0363-1.
45. Jiang N, Englehart KB, Parker PA. Extracting simultaneous and proportional neural control information for multiple-dof prostheses from the surface electromyographic signal. IEEE Trans Biomed Eng. 2009;56:1070–80.
46. Farina D, Vujaklija I, Sartori M, Kapelner T, Negro F, Jiang N, et al. Man/machine interface based on the discharge timings of spinal motor neurons after targeted muscle reinnervation. Nat Biomed Eng. 2017;1:0025.
47. Bengio Y, Courville A, Vincent P. Representation learning: a review and new perspectives. IEEE Trans Pattern Anal Mach Intell. 2013;35:1798–828.
48. Fitts PM. The information capacity of the human motor system in controlling the amplitude of movement. J Exp Psychol. 1954;47:381–91.
49. Caruana R. Multitask learning. Mach Learn. 1997;28:41–75.
50. Rifai S, Vincent P, Muller X, Glorot X, Bengio Y. Contractive auto-encoders: explicit invariance during feature extraction. In: Conference on proceedings of 28th International Conference on Machine Learning (ICML) 2011. 2011. p. 833–840.
51. Maas AL, Hannun AY, Ng AY. Rectifier nonlinearities improve neural network acoustic models. In: Conference on proceedings of 30th International Conference on Machine Learning (ICML) 2013. 2013. p. 1–6.
52. Ba JL, Kiros JR, Hinton GE. Layer normalization. arXiv preprint arXiv: 1607.06450; 2016.
53. Ameri A, Kamavuako EN, Scheme EJ, Englehart KB, Parker PA. Support vector regression for improved real-time, simultaneous myoelectric control. IEEE Trans Neural Syst Rehabil Eng. 2014;22:1198–209.
54. Caruana R. Learning many related tasks at the same time with backpropagation. Adv Neural Inf Process Syst. 1995;7:657–64.
55. Loshchilov I, Hutter F. Decoupled weight decay regularization. arXiv preprint arXiv: 1711.05101; 2017.
56. Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. J Mach Learn Res. 2010;9:249–56.
57. Vincent P, Larochelle H, Lajoie I, Bengio Y, Manzagol PA. Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. J Mach Learn Res. 2010;11:3371–408.
58. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: machine learning in Python. J Mach Learn Res. 2011;12:2825–30.
59. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, et al. TensorFlow: a system for large-scale machine learning. Proc 12th USENIX Symp Oper Syst Des Implementation. 2016;16:265–2283.
60. Proakis JG, Monolakis DG. Digital signal processing: principles, algorithms, and applications. 3rd ed. Upper Saddle River: Prentice Hall; 1996.
61. Farrell TR, Weir RF. The optimal controller delay for myoelectric prostheses. IEEE Trans Neural Syst Rehabil Eng. 2007;15:111–8.
62. Williams MR, Kirsch RF. Evaluation of head orientation and neck muscle EMG signals as command inputs to a human-computer interface for individuals with high tetraplegia. IEEE Trans Neural Syst Rehabil Eng. 2008;16:485–96.
63. Cramer AOJ, van Ravenzwaaij D, Matzke D, Steingroever H, Wetzels R, Grasman RPPP, et al. Hidden multiplicity in exploratory multiway ANOVA: prevalence and remedies. Psychon Bull Rev. 2016;23:640–7.
64. Cohen J. Statistical power analysis for the behavioural science (2nd Edition). 2nd ed. Hillsdalle: Erlbaum; 1988.
65. Young AJ, Hargrove LJ, Kuiken TA. The effects of electrode size and orientation on the sensitivity of myoelectric pattern recognition systems to electrode shift. IEEE Trans Biomed Eng. 2011;58:2537–44.
66. Bunderson NE, Kuiken TA. Quantification of feature space changes with experience during electromyogram pattern recognition control. IEEE Trans Neural Syst Rehabil Eng. 2012;20:239–46.
67. Powell MA, Kaliki RR, Thakor NV. User training for pattern recognition-based myoelectric prostheses: Improving phantom limb movement consistency and distinguishability. IEEE Trans Neural Syst Rehabil Eng. 2014;22:522–32.
68. Krasoulis A, Vijayakumar S, Nazarpour K. Effect of user practice on prosthetic finger control with an intuitive myoelectric decoder. Front Neurosci. 2019;13:891.
69. Patel GK, Nowak M, Castellini C. Exploiting knowledge composition to improve real-life hand prosthetic control. IEEE Trans Neural Syst Rehabil Eng. 2017;25:967–75.
70. González DS, Castellini C. A realistic implementation of ultrasound imaging as a human-machine interface for upper-limb amputees. Front Neurorobot. 2013;7:17.

Publisher's Note

Paper IV

A database of high-density surface electromyogram signals comprising 65 isometric hand gestures

N. Malešević, **A. Olsson**, P. Sager, E. Andersson, C. Cipriani, M. Controzzi, A. Björkman, C. Antfolk

Published in: Scientific Data 8, 63 (2021)

SCIENTIFIC DATA

OPEN

DATA DESCRIPTOR

A database of high-density surface electromyogram signals comprising 65 isometric hand gestures

Nebojša Malešević[1]✉, Alexander Olsson[1], Paulina Sager[1], Elin Andersson[1], Christian Cipriani[2,3], Marco Controzzi[2,3], Anders Björkman[4] & Christian Antfolk[1]✉

Control of contemporary, multi-joint prosthetic hands is commonly realized by using electromyographic signals from the muscles remaining after amputation at the forearm level. Although this principle is trying to imitate the natural control structure where muscles control the joints of the hand, in practice, myoelectric control provides only basic hand functions to an amputee using a dexterous prosthesis. This study aims to provide an annotated database of high-density surface electromyographic signals to aid the efforts of designing robust and versatile electromyographic control interfaces for prosthetic hands. The electromyographic signals were recorded using 128 channels within two electrode grids positioned on the forearms of 20 able-bodied volunteers. The participants performed 65 different hand gestures in an isometric manner. The hand movements were strictly timed using an automated recording protocol which also synchronously recorded the electromyographic signals and hand joint forces. To assess the quality of the recorded signals several quantitative assessments were performed, such as frequency content analysis, channel crosstalk, and the detection of poor skin-electrode contacts.

Background & Summary

The electromyographic signal (EMG) encodes information related to the recruitment patterns of motor neurons innervating skeletal muscles close to the site of signal acquisition. A good understanding of the underlying electrophysiology is important for studying human biomechanics[1] and for diagnosing neuromuscular disease[2]. Furthermore, the knowledge of the underlying neurophysiology behind motor control acquired non-invasively by surface EMG (sEMG) and high density surface EMG (HD-sEMG) has attracted increasing interest in the pursuit of novel human-computer interfaces[3]. Among salient uses for this application of the technique in the field of upper-limb prosthetics, where sufficiently accurate mappings from measured forearm myoelectricity to hand- and wrist kinematics could be used to deliver intuitive motor commands to a robotic replacement limb. At present, commercially available myoelectric prostheses are most commonly controlled via direct, proportional control of a single degree of freedom (DoF)[4]. Advanced multifunctional prostheses[5] are within this framework typically controlled sequentially by employing some protocol for switching between active DoFs[6]. Although simple, this type of interface is often perceived as slow and unintuitive by the user, requiring nontrivial cognitive efforts and leading to a high number of users abandoning their prosthesis[7]. Efforts to improve the ability to automatically decode forearm sEMG into natural movement commands, therefore, have the potential to be of considerable value for transradial (forearm) amputees.

Despite having been the subject of studies for several decades, the exact relationship connecting sEMG to limb kinematics remains in part elusive. Due to the apparent stochasticity, nonstationarity, and nonlinearity of sEMG with respect to muscle contractions[8–10], many studies aimed at the extraction of motor intent have found success by refraining from modelling this relationship explicitly and instead resorting to a combination of manual *feature engineering* and *machine learning*[11]. With this strategy, the information density of acquired sEMG signals is increased by compression into a set of numeric descriptors (i.e. features) via a sliding time window technique. Given that the selected features capture discriminative properties of the latent generative process, pattern

[1]Department of Biomedical Engineering, Faculty of Engineering, Lund University, Lund, Sweden. [2]The BioRobotics Institute, Scuola Superiore Sant'Anna, Pisa, Italy. [3]Department of Excellence in Robotics and AI, Scuola Superiore Sant'Anna, Pisa, Italy. [4]Department of Hand Surgery, Clinical Sciences, Sahlgrenska Academy, University of Gothenburg and Sahlgrenska University Hospital, Gothenburg, Sweden. ✉e-mail: nebojsa.malesevic@bme.lth.se; christian.antfolk@bme.lth.se

recognition algorithms[12–15] can thereafter be deployed to map such features to grasps and motions. More recently, the application of automated *feature learning* in the form of deep neural networks[16–23] has enabled end-to-end mappings directly, from raw EMG to movement representations. Such methods are arguably uniquely appropriate for processing HD-sEMG due to the signal's structural resemblance to conventional image data. Both feature engineering and deep learning circumvent the complexity of explicitly modelling the underlying electrophysiology, but at the cost of requiring labelled data sets containing (I) recorded EMG and (II) numeric representation of synchronous performed movements.

Multiple databases[18,24–29] exist which contain collections of EMG or HD-sEMG together with synchronous movement stimuli and/or joint angle time series. The general importance of publicly available resources of this kind can be understood as twofold: firstly, it allows for the development of novel methods by other researchers without requiring time-consuming data collection. Secondly, it allows for inter-method collation, as the performances of different methods are difficult to compare fairly if evaluated on data sets with differing characteristics.

The aim of this data set was to contribute to the development of better myoelectric decoding schemes by presenting a new HD-sEMG data set, distinguished by a different approach compared to previous contributions to the same end. 128 channels of sEMG data were recorded at the level of the forearm from 20 able-bodied and healthy participants with a recording protocol constituted by 65 unique movements. These 65 movements were furthermore interpreted as compounds of 16 basis movements that capture the major DoFs of the hand and wrist. It is our hope that this type of decomposition will allow for multi-label machine learning[20] approaches to be leveraged and potentially lead to the development of more dextrous control interfaces. Furthermore, forces exerted at the level of the wrist and the digits were collected and are provided here to allow for regression-type approaches, which might offer new possibilities in the domain of proportional control for prosthetic hands.

Methods

Participants. Twenty able-bodied volunteers (14 men and 6 women) aged between 25 and 57 years (mean age 35 years) participated in the study. All participants were right-handed and neurologically intact. All participants provided informed consent, and the study was approved by the Regional Ethical Review Board in Lund, Sweden (Dnr 2017-297).

High density sEMG recording. The EMG signals were recorded using a Quattrocento (OT Bioelettronica, Torino, Italia) biomedical amplifier system. The Quattrocento is able to acquire up to 400 channels sampled with 16-bit resolution. In this study, the EMG recording chain comprised high density sEMG electrodes, preamplifiers with 5x gain located at the electrode connectors, amplifiers within the Quattrocento device, and the A/D converters. In total, including preamplifiers and amplifiers, the HD-sEMG signals were amplified 150 times. The EMG signals were sampled at 2048 Hz and a hardware high-pass filter at 10 Hz and a low-pass filter at 900 Hz were used during recordings.

The electrodes used in his study consisted of 64 contacts arranged in an 8×8 matrix, with an inter-electrode distance of 10 mm (ELSCH064NM3, OT Bioelettronica, Torino, Italy). To reduce common-mode noise in the EMG signal, the recording was performed in a differential manner. In this mode, consecutive channels were subtracted, where the enumeration of the electrode channels is shown in Fig. 1c. Due to the electrode orientation with respect to the underlying muscles, the differentiation of the EMG signals was done along the muscle fibers. As the result, ch1 signal was calculated as the difference between EMG signals at electrode contacts 2 and 1, ch2 as the difference between signals at contacts 3 and 2, and so on. This methodology also implies that channels that are a multiple of 8 (ch8, ch16…) have different EMG signal pick-up area as contacts 8 and 9 (contacts 16 and 17, and so on) span along the whole length of the electrode. Additionally, the last channel of the electrode (ch64) is calculated as the difference between EMG signals at first contact of the next electrode (contact 1) and the last electrode contact (contact 64) of the current electrode.

Two HD-sEMG electrodes were positioned on the dorsal and the volar aspects of the forearm with the intention to cover, or partially cover, the main fingers flexors and extensors (flexor digitorum profundus – responsible for flexion of fingers D2-D5, extensor digitorum communis – responsible for extension of fingers D2-D5), wrist flexor/extensor (flexor carpi radialis, flexor carpi ulnaris – responsible for wrist flexion, extensor carpi radialis longus, extensor carpi ulnaris – responsible for wrist extension) and pronator/supinator (pronator teres, supinator), and thumb flexor/extensor (flexor pollicis longus – responsible for thumb flexion, extensor pollicis longus – responsible for thumb extension) and thumb abduction (abductor pollicis longus). As the HD-sEMG electrodes can cover a relatively large area, the positioning of the electrodes was guided by physiological landmarks, such as distance from the elbow for the distal placement, and distance from the ulna for radial orientation. The electrodes were placed approximately 3 cm from elbow (elbow to closest electrode corner) and 2 cm from ulna (edge of the ulna to edge of the electrode). The positions of the electrodes are shown in Fig. 1. An electrode consists of a thin, flexible substrate layer on which a foamy single-use double-adhesive layer is applied. The foamy layer has holes punched at the locations of electrode contacts which were filled with a conductive gel. This structure permits a tight and comfortable fit on a forearm regardless of the circumference. In addition, to ensure firm electrode contact throughout a long measurement (lasting approximately 1 h) an elastic bandage was placed over both electrodes. The reference electrode, which was in the form of a ribbon, was placed around the wrist.

The HD-sEMG signals were recorded with the OT Biolab program (OT Bioelettronica, Torino, Italia) that saves the uncompressed data in a proprietary file format.

Isometric force recording. A custom-made force measurement device was used to obtain hand forces during the recording protocol[30]. The device was designed in a manner that enables independent acquisition of finger and wrist isometric forces. The motivation for choosing an isometric setup was to simulate muscle behaviour in a

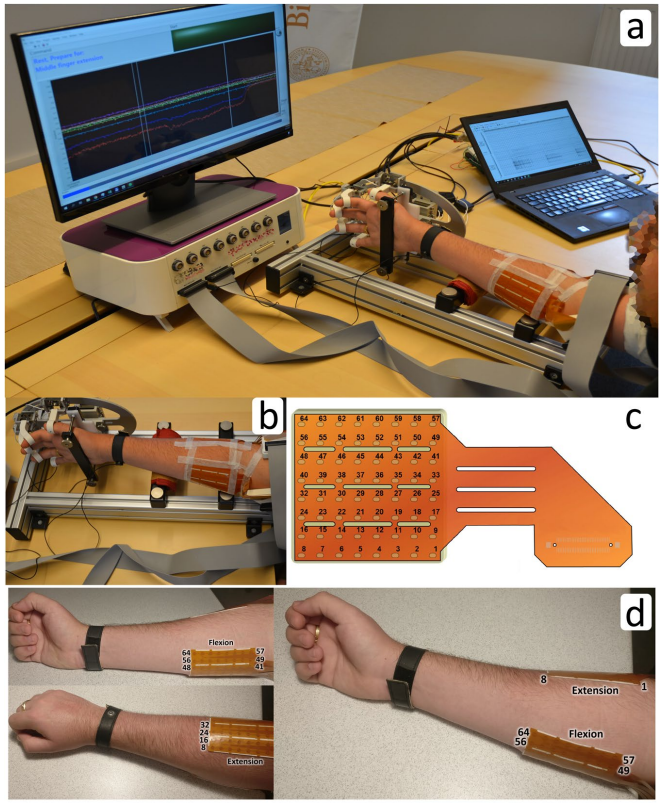


Fig. 1 Measurement setup. (**a**) The participant's hand was positioned inside the force measurement device. Two 64 contacts electrodes were placed on dorsal and volar aspects of the upper forearm. Preamplifiers were placed at the electrode connector and amplified EMG signals were routed to Quattrocento device. HD-sEMG signals were displayed on a laptop screen in real-time while hand forces and cues were shown on a separate screen in front of the participant. (**b**) Hand positioned inside force measurement device taken from a different angle. (**c**) Pinout of ELSCH064NM3 electrode (skin top view). The differentiation of the EMG channels was done along the increasing channel indexes. For example, the first output (ch1) is calculated as the difference between EMG signals at contacts 2 and 1, ch8 as 9-8, and the last output (ch64) as the difference between the first of the next electrode and the last of the current electrode. (**d**) Position of flexion and extension electrodes in supinated and fully pronated forearm orientations.

forearm amputee where the remaining muscles have a relatively small contraction amplitude. The hand position inside the force measurement device is shown in Fig. 1.

The device comprises nine strain gauges, four measuring D2-D5 flexion/extension forces, two measuring thumb flexion/extension and abduction/adduction, and three measuring wrist flexion/extension, pronation/supination, and radial/ulnar deviation. Force gauges and hand joints were interfaced through 3D printed finger braces which were specifically chosen for each participant. Using the braces, the fingers were placed in a neutral position, approximately in the middle of the range of motion. During the recording protocol, the force measurement device was placed on the table with the participant sitting in front of it. The chair height was adjusted to provide the participant with a comfortable body posture during the recordings. Although the device could be adjusted for both, right and left hand, for simplicity in this study it was used only in the right-hand setup (as all participants were right-handed).

The measured hand forces were provided as analog signals in the range of 0–5 V for the force range ± 100 N, where in neutral position (0 N) the force sensors value was 2.5 V. Within this range, the sensor output was proportional to the force with less than 1% full-scale error. The signals from the force sensors were digitalized using a NI-USB 6218 (National Instruments, Austin, Texas, USA) A/D with 16-bit amplitude resolution and sampling

frequency of 200 Hz. The visualization and the recording of the signals were managed by a custom-made LabVIEW (National Instruments, Austin, Texas, USA) program. The same program controlled synchronization between the Quattrocento and the force signals by generating TTL pulses recorded by both devices. The pulses generated by one of the digital outputs of NI-USB 6218 were 0.2 s wide and occurring every 2 s. The forces presented in this paper are given in volts [V], and the transfer function between sensor analog output and the force is the following:

force = analog_voltage*40–100 [N]

The detailed description of the measurement device and its error validation could be found in our previous publication[30].

Recording protocol. The recording session was initiated by a brief explanation of the protocol to the participant, after which he/she signed the informed consent. Next, the chair height and finger braces were adjusted to fit the participant to ensure comfortable body posture and hand fit inside the force measurement device. Before applying the electrodes, the participant was informed about the specific hand movements to be performed during recordings. The list of all the movements was provided to the participant, and a time period was given to the participant to go through the list and try to execute the hand movements with the hand outside the force measurement device. In the case the participant felt that a specific hand movement was difficult to execute, the participant had an opportunity to ask for an explanation and practice the movement in both an isometric manner and in a free-hand manner. Upon confirming that the participant was able to perform the hand movements from the list, the electrodes were placed on the upper forearm. Subsequently, the participant was asked to place the hand into the force measurement device and perform random hand movements so that the electrode-skin contact could be assessed. At this stage, it was checked if there were any EMG channels containing high noise or spikes that occurred during the muscle contractions. The presence of a high amplitude signal was usually an indicator that some of the electrodes have poor contact with the skin surface. In these cases, the elastic bandage covering the electrodes was tightened. If there were still channels with high amplitude noise, the electrode was removed and reapplied at the same position (with a new foamy layer and gel). Upon confirming that the EMG channels were not contaminated with excessive noise nor movement artefacts the automated measurement protocol was started.

The recording protocol consisted of 66 hand movements (65 unique movements and one repeated twice) with five repetitions separated with rest periods each lasting 5 s. As the main aim of this database was to provide useful data for EMG classification, specifically for multi-label classification, the selection of movements was made so that it comprised all single degree of freedom movements (1DoF) that could be obtained by the means of the force measurement device, such as flexion/extension of individual fingers, but also, compound movements comprising combinations of basic movements. In this study, 16 basic/1DoF movements were selected, two per D2-D5 fingers (flexion-extension), four for thumb (flexion-extension-abduction-adduction) and four for wrist (flexion-extension-pronation-supination). To make the movement commands more understandable for the participants, terms flexion-extension for D2-D5 fingers were replaced with bend-stretch, flexion-extension-abduction-adduction for thumb with down-up-left-right, and flexion-extension-pronation-supination for wrist with bend-stretch-rotate anti clockwise-rotate clockwise. With 16 basic movements, the number of all the possible combinations is very high, and it would be impractical to record all of them. Thus, the subset of compound movements was derived using several rules:

1. All co-contractions of a single joint were excluded from the list. For example, a command to simultaneously flex and extend a finger was not included as the net finger force would be zero, thus it would not be detected by the force sensor.
2. For movements comprising two fingers, only flexions of adjacent fingers were included in the list. Besides being more common in activities of daily living, the flexions of adjacent fingers are movements that could be performed without specific motor skills or extensive training (unlike flexion of non-adjacent fingers).
3. Hand movements including combinations of joints flexion and extension were excluded from the list, with the exception of wrist extension that was included with fingers flexions, and pointing movement that included extension of index finger together with flexion of D3-D5 fingers. Similar to the rationale provided for rule 2, hand movements comprising a mixture of joints flexions and extensions are rare in activities of daily living, and difficult to execute.
4. For the multi-joint hand movements (>2 DoF movements), the selection was based on the most common hand grips and gestures. These movements include known muscle synergies that enable easy and intuitive execution even in an isometric fashion.
5. As the fingers extensions were underrepresented in the recording set, two instances of D2-D5 fingers extensions were included in the list of movements (movement codes 58 and 60 in Online-only Table 1).

During the pilot measurements (not included in this study), the list of movements was modified and some of the hand movements were removed if there were difficulties in performing them. In total, 66 hand movements (65 unique) were used in this protocol, see Online-only Table 1.

The measurement was guided automatically by the custom-made software developed in LabVIEW. The graphical interface presented to the participant textual commands for the next/current hand movement (as in Online-only Table 1). The onset of the hand movement was directed by a large green light indicator (visual cue) and a short beeping sound (auditory cue). The participant was supposed to "hold" the movement at a comfortable force level until the visual indicator was turned off. The movement end was also signalled by the change of the displayed textual command, which during the rest periods comprised words "Rest, prepare for: (movement from Online-only Table 1)". Each hand movement was performed five times before switching to the next movement. The transition between different movements was additionally highlighted by the change of the text colour, which

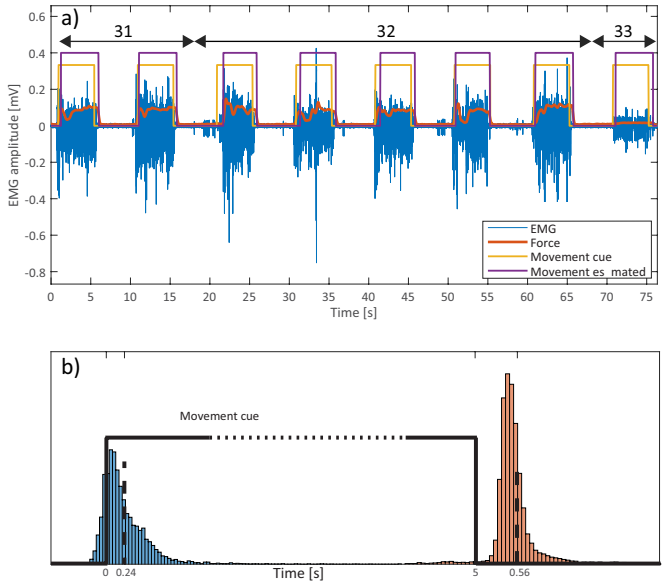


Fig. 2 An example of the recorded signals, markings, and histograms of latencies for movement onsets and cessations. (**a**) shows an EMG channel (out of 128) from the electrode placed over the volar portion of the forearm. Superimposed with the EMG signal (blue) are ring finger force (red), movement cue timings presented to the participant (yellow), and the re-labelled movement durations estimated using force levels (purple). This signal example comprised movements 31 (Ring finger: bend + Wrist: rotate anti-clockwise), 32 (Ring finger: bend + Wrist: rotate clockwise) and 33 (Middle finger: bend + Index finger: bend). From the figure, it could be noted how the force fluctuates during movements, similarities, and differences between consecutive repetitions of the same movement, but also clear differences between the EMG and the force profiles when switching to a completely different finger (joint). (**b**) shows latencies of re-labelled movements for all repetitions, movements, and participants. Mean latencies for onset was 0.24 s, while for the cessation was 0.7 s.

toggled between red and blue after the fifth repetition of a movement. This feature was added as it was noted during the pilot recordings that the participants tend to focus more on the movement onset cue than on the textual command. This often resulted in producing the same hand movement for extra repetitions although the movement textual command was changed. With the colour change between different hand movements, the occurrence of wrong repetitions was significantly reduced.

Data processing. The HD-sEMG data recorded with OT Biolab was filtered offline to further remove power line noise. A zero-phase 3rd order band-pass Butterworth filter centered at 50 Hz with 4 Hz width implemented in Matlab (command: *filtfilt*) was used for this task. No additional processing was done to the HD-sEMG signals.

The signals acquired with two devices (Quattrocento and NI-USB 6218) and recorded with two programs running in parallel (OT Biolab and LabVIEW) were synchronized using the common TTL pulses that were supplied to both recording chains. In the offline processing, these pulses were detected in both files and the signals were truncated so that the beginning was at the leading edge of the first pulse, while the end was at the trailing edge of the last pulse. As an additional check, the pulses were counted in both files to verify that there was no missing data. To join the HD-sEMG data sampled at 2048 Hz and force and movement label data sampled at 200 Hz, interpolations were performed so that all the signals matched the HD-sEMG sampling rate (2048 Hz). This step was done using Matlab command *interp1*.

The onset and cessation of each movement were presented to the participant using visual and audio cues. Nevertheless, the hand movement was usually delayed due to physiological reaction time, but also due to reduced focus of the participant during prolonged measurement. Thus, the real movement onset and cessation were not matching the movement labels that correspond to movement cues. To provide movement labels that correspond to real hand movements, a temporal re-labelling was done (see Fig. 2a). The main aim of this process was to provide movement (class) labels that separate rest periods from movements, thus providing consistent signals for training procedure of a classifier, but also providing better segmented data for classifier testing purposes. The temporal re-labeling relied on measured hand forces as the indicators of the movement offset and cessation. The algorithm of temporal re-labeling can be described as the sequence of the following steps:

- In the period preceding the movement cue, the "rest" hand forces were obtained as the average values within a 1 s window. This resulted in a 1×9 vectors (one for each sensor) that was updated before each movement cue.
- The "rest" hand forces were subtracted from each force channel. This process was done in a loop that stepped through the signal in a sample-by-sample manner, where the "rest" vector was updated at the beginning of each movement cue.
- Subtracted signals were rectified and all 9 force channels were summed together for each sample, resulting in a 1 x (signal_length) vector.
- For each hand movement, the algorithm found the minimal hand force within 1 s period preceding the movement onset cue, and maximal hand force during 5 s after the movement onset cue.
- The real movement onset was then defined as the sample at which the summed force signal was above 50% of the difference between minimal and maximal movement forces.
- A similar procedure was done for estimating when the real hand movement ended. For each hand movement, the algorithm found the maximal hand force within 1 s period preceding the movement cessation cue, and minimal hand force during the 5 s after the movement cessation cue.
- The real movement cessation was then defined as the sample at which the summed force signal crosses below 50% of the difference between minimal and maximal movement forces.

Calculated latencies between the presented cue and the real movement were joined for all participants, movements, and repetitions (Fig. 2b). Based on the cumulative data, mean latencies were extracted; 0.24 s (std = 0.33 s) for onset and 0.56 s (std = 0.46 s) for cessation. It should be noted that movement onset was expressed by both auditory and visual cue, while movement cessation was only expressed by turning off of the visual cue (onscreen virtual LED). This fact could be used to explain the differences between latency distributions for onset and cessation of the movements. Another difference between these two conditions is that for onset, participants were focusing on the incoming cues, while for the movement cessations, the participants were focusing on maintaining the desired hand gesture, and transitioning to rest state required more time. Finally, as the movement executions were rhythmical (5 s on, 5 s off, 5 s on…) it was possible to anticipate the timing of the onset cue, which is also the reason for some early movement onsets (before actual cue). The observed movement latencies (relabelling results) are comparable with previous studies focused on human reaction time[31,32].

Data Records

The data presented in the current paper can be downloaded from figshare[33] and used freely for any purpose. This section describes the contents of the provided files.

Each file, encoded in.*mat* format, contains data recorded from a single test participant and is named according to the convention *sx.mat*, where x is the participant index (i.e. an integer in the range 1–20). Each file contains a set of variables, listed below, together constituting the data and metadata connected to the participant. *L* here denotes the total number of sampled time points across the entirety of the recording session.

- *subject*: An integer in the range 1–20, representing the participant number (same as in file name).
- *Fs*: The sampling rate of the HD-sEMG signal and of the synchronized forces and movement stimuli signals, here always equal to 2048 Hz.
- *emg_extensors*: An $L \times 8 \times 8$ matrix containing the sEMG samples of the 64 channels recorded from the dorsal side of the forearm. The second and third matrix indices correspond to the relative positions of the corresponding electrodes along and across the forearm, respectively.
- *emg_flexors*: An $L \times 8 \times 8$ matrix containing the sEMG samples of the 64 channels recorded from the volar side of the forearm. The second and third matrix indices correspond to the relative positions of the corresponding electrodes along and across the forearm, respectively.
- *force*: A $L \times 9$ matrix containing the 6 force channels, samples synchronized with those of the EMG signals.
- *class*: An 1D array of length L containing integers in the range [0,65] which encodes the class of the movement stimuli being presented to the participant concurrently with collected sEMG and forces. Array elements of value 0 denote the rest state.
- *labels*: An $L \times 16$ Boolean matrix, computed directly from the *class* variable via a lookup table. The truth value of *labels*(i, j) is 1 if the movement which the participant is prompted to perform at time j incorporates the i:th DoF and 0 if it does not.
- *repetition*: An 1D array of length L containing integers in the range 0–5 which encodes the repetition number of the movement stimuli being presented to the participant.
- *adjusted_class*: The *class* variable following automated re-labelling as described in the Data Processing section.
- *adjusted_labels*: The *labels* variable following automated re-labelling as described in the Data Processing section.
- *adjusted_repetition*: The repetition variable following automated re-labelling as described in the Data Processing section.
- *outlier_scores_extensors*: An 8×8 matrix containing channel-specific outlier scores, computed via the method described in the Technical Validation section, of the sEMG channels presented in *emg_extensors*. The value stored in *outlier_scores_extensors[i, j]* corresponds to the channel *emg_extensors[i, j, :]*.
- *outlier_scores_flexors*: An 8×8 matrix containing channel-specific outlier scores, computed via the method described in the Technical Validation section, of the sEMG channels presented in *emg_flexors*. The value stored in *outlier_scores_flexors[i, j]* corresponds to the channel *emg_flexors[i, j, :]*.

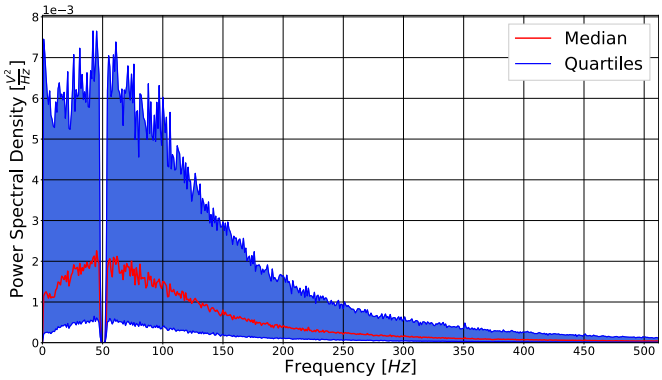


Fig. 3 Aggregated representation of HD-sEMG signal spectra from all participants, movements, and channels. The central red line represents the median spectrum and the blue regions represent the quartiles (computed separately for each frequency bin).

Technical Validation

In addition to the protocol followed during recording sessions for ascertaining the quality of acquired signals (described in the Methods section), the resulting dataset was subsequently assessed quantitatively in an offline setting. This validation of HD-sEMG signals entailed (i) extracting signal frequency spectrums from all EMG channels, (ii) computing the cross-correlations between all possible pairs of channels for all recording sessions, and (iii) computing a channel outlier metric separately for each participant and EMG channel. The details and implications of these approaches are given in the following sections.

Frequency spectra. The frequency spectra of the EMG signals comprising the current dataset are summarized graphically in Fig. 3. A single spectrum was computed for each participant and channel (20 128 = 2560 spectrograms in total) using Welsh's method; the curves plotted in Fig. 3 represents quartiles, computed frequency-wise, over all such spectra. Excluding the discontinuities induced by offline notch filtering at 50 Hz power line interference, the morphologies of the spectra correspond to those expected in light of the previous studies[34]. As it could be expected, there was a notable variation in average amplitude between deciles, which is a result of the variation in average amplitude between channels – the 128 electrodes cover muscles situated at different depths, and different muscles are moreover recruited for different numbers of unique movement classes, resulting in significant variation in amplitude and by extension observable variations in vertical offsets of deciles.

Channel correlations. The two HD-sEMG electrode arrays used for the current study were, during the experiments, placed on the skin in a way to cover multiple forearm muscles. Consequently, when a participant attempted to perform a movement, a specific subset of covered muscles was recruited, leading to a spatially clustered pattern of activity in the concurrent HD-sEMG. This behavior is the consequence of the nature of the electromyography technique that relies on the electrical signal traversing along the muscle fibers (action potential) which is then transmitted through tissues radially, eventually reaching recording electrodes. The amplitude of this signal on different electrodes is proportional to the distance between the source (muscle) and the recording points (electrodes). The zero-lag cross-correlation coefficient between a given pair of EMG channels is therefore expected to be large for channels with electrodes situated close together, and small for channels with electrodes situated far apart. In the event of non-negligible interference or other types of noise shared across multiple channels (e.g. excessive motion artefacts), this regularity can no longer be expected to hold true, as even signals acquired by electrodes far apart would exhibit notable covariation. To verify the absence of this type of noise in the presented dataset, the zero-lag cross-correlation coefficient between every possible pair of channels was computed for each participant and compared to the physical distance separating the electrodes of the pair. The analysis was performed on whole signals (comprising all movements) for all electrode pairs within the same row or column, and for all the participants. Only pairings where both channels belong to the same electrode row/column were used. This also means that only channels within the same electrode were considered, as the distance separating the two 8×8 electrode arrays was not noted. As the electrodes were coarsely aligned with the direction of muscle fibers of major muscles within the forearm, the electrical activity picked by electrode rows (channels 1-2-3… 63–64) and columns (channels 1-9-17-… 56–64) is resulting from different physical processes. In the case of electrode rows, the electrical signal is directly coupled with the propagation of action potentials along the muscle fibers, while in the case of electrode columns the signal is reaching recording sites by passive radial propagation from muscle fibers through surrounding tissues. To observe both of these effects the cross-correlation was calculated for the two directions separately and is presented in Fig. 4. In Fig. 4a, the observed relationship of the inter-electrode distance across muscles and channel cross-correlation is presented; Fig. 4b contains the observed relationship of the distance along muscles and channel cross-correlation. In both

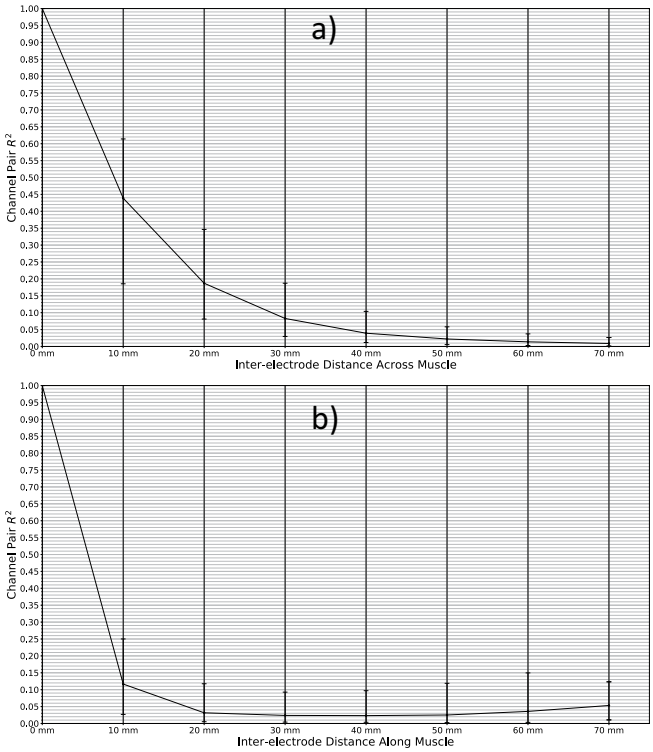


Fig. 4 Coefficient of determination R^2 of channels pairs for all 8 possible inter-electrode distances **(a)** perpendicular to muscles and **(b)** parallel to muscles. The central markers represent means and the upper and lower delimiters of the error bars represent the 75th and 25th percentile, respectively, computed across all participants and possible channel pairings.

cases, a strong negative relationship was observed between the variables, indicating that the amount of common noise and crosstalk was limited, but it is notable that the channel cross-correlation decreases much slower in the case of electrodes placed across the muscle. As mentioned before, this behavior reflects underlying electrophysiology coupled with the zero-lag cross-correlation method which was used to assess channels crosstalk. The employed method results in (1) higher cross-correlation values between channels across the muscle where the signal from muscle fibers is only attenuated in proportion to the electrode-fiber distance (Fig. 4a), and (2) lower cross-correlation when the signal is propagated with some time delay (Fig. 4b).

Outlier scores. As the used HD-sEMG electrodes have some mechanical constraints, such as limited curvature and adhesion properties, there is always a possibility of having a poor skin-electrode contact at specific electrode portions. This issue results in an increased environmental noise level (50 Hz) and the appearance of movement artifacts in the form of high-amplitude spikes during some specific contractions that deform the forearm surface more than the electrode can follow. Thus, the signal quality of HD-sEMG can plausibly be expected to vary across channels for a given participant and recording session. Although the movement artifacts spikes are usually very sparse, as they appear together with some specific hand gestures, they could potentially, impede some of the machine-learning algorithms by providing false EMG behavior.

In subsequent offline processing of signals, it may thus be of interest to exclude channels deemed as *outliers* by some appropriate measure of channel deviation. In order to provide such a measure in the current dataset, a simple *outlier score* was defined and calculated for each individual channel and recording session. To calculate this score, denoted O_i for the i^{th} channel, the following procedure was carried out: initially, the 99th percentile of all rectified voltages sampled from each channel, denoted $p_i^{99\%}$. for the i^{th} channel, was calculated. This value approximates the voltages reached by the considered channel at signal peaks. This statistical measure proved to be more robust than simply finding the maximum voltage across all samples. Next, the first and third quartile (Q1 and Q3, respectively) were extracted from the lists of all such values across all channels ($p_i^{99\%}$ for all i) and used to compute

the interchannel *interquartile range* as $IQR = Q3 - Q1$. In accordance with one common definition of a statistical outlier, a nonzero outlier score O_i was lastly assigned to the i^{th} EMG channel if and only if $p_i^{99\%}$ exceeded a threshold voltage given by $T = Q3 + 1.5 \cdot IQR$. For such channels, i.e. where $p_i^{99\%} > T$, the value of O_i was set to be proportional to the number of IQRs with which $p_i^{99\%}$ exceeded the threshold:

$$O_i = \frac{\max(0,\ p_i^{99\%} - T)}{IQR}$$

A channel with an outlier score $O_i = 0$ can be interpreted as falling within expected boundaries of valid EMG amplitude variation. For a channel with nonzero O_i, the outlier score is intended to quantify the degree to which the channel generated notably higher peak voltages, as caused by e.g. signal high-amplitude spiking, than those of the other channels. With the provided list of scores, it is possible to exclude channels at an arbitrary level of acceptable channel deviation.

Due to the bipolar sampling protocol used to acquire EMG signals during recording sessions, an outlier score was computed only for the $128 - (8 \cdot 2) = 114$ channels not originating from the electrodes at the proximal end of the two electrode arrays (the remaining 16 channels were automatically given an outlier score of 0). In the current database, recording sessions contained an average of 92.24% (SD 3.61%) channels with an outlier score of 0. Among channels with nonzero outlier score, the mean value of O_i was calculated as 2.63 (SD 3.68).

In addition, due to the bipolar setup, channel 128 (the last channel of the second electrode) should not be used as it was not referenced in the same manner as other channels.

Code availability

The signal recording was performed using two programs in parallel: OT BioLab version 2.0.6254 available at www.otbioelecttronica.com for recording HD-sEMG and synchronization signals, and the custom recording software developed in LabVIEW 2016 for force signals recording, generating synchronization pulses, visualizing forces, and generating commands and cues. Data post-processing was done in Matlab and Python. The custom codes for temporal re-labeling and outlier scores are available at the GitHub repository: https://github.com/Neuroengineering-LTH/HDsEMG-database-Associated-codes.

Received: 20 August 2020; Accepted: 19 January 2021;
Published: 18 February 2021

References

1. Bonato, P., Boissy, P., Della Croce, U. & Roy, S. H. Changes in the surface EMG signal and the biomechanics of motion during a repetitive lifting task. *IEEE Trans. Neural Syst. Rehabil. Eng.* **10**, 38–47 (2002).
2. Wimalaratna, H. S. K., Tooley, M. A., Churchill, E., Preece, A. W. & Morgan, H. M. Quantitative surface EMG in the diagnosis of neuromuscular disorders. *Electromyogr. Clin. Neurophysiol.* **42**, 167–172 (2002).
3. Ahsan, M. R., Ibrahimy, M. I. & Khalifa, O. O. EMG signal classification for human computer interaction: A review. *European Journal of Scientific Research* **33**, 480–501 (2009).
4. Atkins, D. J., Heard, D. C. Y. & Donovan, W. H. Epidemiologic overview of individuals with upper-limb loss and their reported research priorities. *Journal of Prosthetics and Orthotics* **8**, 2–11 (1996).
5. Saikia, A. *et al.* Recent advancements in prosthetic hand technology. *Journal of Medical Engineering and Technology* **40**, 255–264 (2016).
6. Fougner, A., Stavdahl, O., Kyberd, P. J., Losier, Y. G. & Parker, P. A. Control of upper limb prostheses: Terminology and proportional myoelectric controla review. *IEEE Trans. Neural Syst. Rehabil. Eng.* **20**, 663–677 (2012).
7. Biddiss, E. & Chau, T. Upper-limb prosthetics: Critical factors in device abandonment. *Am. J. Phys. Med. Rehabil.* **86**, 977–987 (2007).
8. Farina, D. & Merletti, R. Comparison of algorithms for estimation of EMG variables during voluntary isometric contractions. *J. Electromyogr. Kinesiol.* **10**, 337–349 (2000).
9. Norali, A. & Som, M. Surface Electromyography Signal Processing and Application: A Review. In *International Conference on Man-Machine Systems* (2009).
10. Goen, A. & Tiwari, D. C. Review of Surface Electromyogram Signals: Its Analysis and Applications. *Int. J. Electr. Electron. Sci. Eng.* **7**, 1–9 (2013).
11. Hudgins, B., Parker, P. & Scott, R. N. A New Strategy for Multifunction Myoelectric Control. *IEEE Trans. Biomed. Eng.* **40**, 82–94 (1993).
12. Hakonen, M., Piitulainen, H. & Visala, A. Current state of digital signal processing in myoelectric interfaces and related applications. *Biomed. Signal Process. Control* **18**, 334–359 (2015).
13. Scheme, E. & Englehart, K. Electromyogram pattern recognition for control of powered upper-limb prostheses: State of the art and challenges for clinical use. *J. Rehabil. Res. Dev.* **48**, 643–660 (2011).
14. Zecca, M., Micera, S., Carrozza, M. C. & Dario, P. Control of Multifunctional Prosthetic Hands by Processing the Electromyographic Signal. *Critical reviews in biomedical engineering* **45**, 383–410 (2017).
15. Malešević, N. *et al.* Vector Autoregressive Hierarchical Hidden Markov Models for Extracting Finger Movements Using Multichannel Surface EMG Signals. *Complexity* **2018**, (2018).
16. Atzori, M., Cognolato, M. & Müller, H. Deep learning with convolutional neural networks applied to electromyography data: A resource for the classification of movements for prosthetic hands. *Front. Neurorobot.* **10**, 1–10 (2016).
17. Geng, W. *et al.* Gesture recognition by instantaneous surface EMG images. *Sci. Rep.* **6** (2016).
18. Du, Y., Jin, W., Wei, W., Hu, Y. & Geng, W. Surface EMG-based inter-session gesture recognition enhanced by deep domain adaptation. *Sensors (Switzerland)* **17**, 6–9 (2017).
19. Ameri, A., Akhaee, M. A., Scheme, E. & Englehart, K. Real-time, simultaneous myoelectric control using a convolutional neural network. *PLoS One* **13**, e0203835 (2018).
20. Olsson, A. E. *et al.* Extraction of Multi-Labelled Movement Information from the Raw HD-sEMG Image with Time-Domain Depth. *Sci. Rep.* **9** (2019).
21. Geng, W. *et al.* A novel attention-based hybrid CNN-RNN architecture for sEMG-based gesture recognition. *PLoS One* **13** (2018).
22. Phinyomark, A. & Scheme, E. EMG pattern recognition in the era of big data and deep learning. *Big Data and Cognitive Computing* **2**, 1–27 (2018).

23. Olsson, A. E., Björkman, A. & Antfolk, C. Automatic discovery of resource-restricted Convolutional Neural Network topologies for myoelectric pattern recognition. *Comput. Biol. Med.* **120**, 103723 (2020).
24. Ortiz-Catalan, M., Brånemark, R. & Håkansson, B. BioPatRec: A modular research platform for the control of artificial limbs based on pattern recognition algorithms. *Source Code Biol. Med.* **8** (2013).
25. Atzori, M. *et al.* Electromyography data for non-invasive naturally-controlled robotic hand prostheses. *Sci. Data* **1** (2014).
26. Pizzolato, S. *et al.* Comparison of six electromyography acquisition setups on hand movement classification tasks. *PLoS One* **12** (2017).
27. Amma, C., Krings, T., Böer, J. & Schultz, T. Advancing muscle-computer interfaces with high-density electromyography. *Conference on Human Factors in Computing Systems - Proceedings* **2015-April**, 929–938 (2015).
28. Matran-Fernandez, A., Martínez, I. J. R., Poli, R., Cipriani, C. & Citi, L. Seeds, simultaneous recordings of high-density EMG and finger joint angles during multiple hand movements. *Sci. Data* **6** (2019).
29. Malesevic, N. *et al.* A database of multi-channel intramuscular electromyogram signals during isometric hand muscles contractions. *Sci. Data* **7** (2020).
30. Malešević, N. *et al.* Instrumented platform for assessment of isometric hand muscles contractions. *Meas. Sci. Technol.* **30** (2019).
31. Valls-Solé, J. *et al.* Reaction time and acoustic startle in normal human subjects. *Neurosci. Lett.* **195**, 97–100 (1995).
32. Pascual-leone, A., Brasil-neto, J. P., Valls-solé, J., Cohen, L. G. & Hallett, M. Simple reaction time to focal transcranial magnetic stimulation: Comparison with reaction time to acoustic, visual and somatosensory stimuli. *Brain* **115**, 109–122 (1992).
33. Malešević, N. *et al.* A database of high-density surface electromyogram signals comprising 65 hand gestures performed in an isometric manner. *figshare* https://doi.org/10.6084/m9.figshare.c.5090861 (2021).
34. Basmajian, J. & De Luca, C. J. Description and Analysis of the EMG Signal. In *Muscles alive: their functions revealed by electromyography* 65–100 (1985).

Acknowledgements
This research was supported by the EU-Funded DeTOP Project (EIT-ICT-24-2015, GA no. 687905), the Swedish Research Council (2019-05601), the Promobilia Foundation and Stiftelsen för bistånd till rörelsehindrade i Skåne.

Author contributions
N.M. designed the software for automated recording protocol, designed the study, performed the data collection and analysis of results, and drafted the manuscript. A.O. performed the data collection and analysis of results and drafted the manuscript. P.S. performed data acquisition and parts of the data preprocessing. E.A. performed data acquisition and parts of the data preprocessing. C.C. designed the study. M.C. designed the study. A.B. designed the study and aided in data acquisition. C.A. designed the measurement hardware, designed the study, performed the data collection and drafted the manuscript.

Funding
Open access funding provided by Lund University.

Competing interests
The authors declare no competing interests.

Additional information
Correspondence and requests for materials should be addressed to N.M. or C.A.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Paper v

End-to-end estimation of hand-and wrist forces from raw intramuscular EMG signals using LSTM networks

A. E. Olsson, N. Malešević, A. Björkman, C. Antfolk

Published in: Frontiers in Neuroscience 15, 777329 (2021)

frontiers in Neuroscience

ORIGINAL RESEARCH
published: 17 November 2021
doi: 10.3389/fnins.2021.777329

End-to-End Estimation of Hand- and Wrist Forces From Raw Intramuscular EMG Signals Using LSTM Networks

Alexander E. Olsson[1], Nebojša Malešević[1], Anders Björkman[2,3,4] and Christian Antfolk[1]**

[1] Department of Biomedical Engineering, Faculty of Engineering, Lund University, Lund, Sweden, [2] Department of Hand Surgery, Sahlgrenska University Hospital, Institute of Clinical Sciences, University of Gothenburg, Sahlgrenska Academy, Gothenburg, Sweden, [3] Department of Hand Surgery, Skåne University Hospital, Malmö, Sweden, [4] Department of Translational Medicine, Lund University, Lund, Sweden

Processing myoelectrical activity in the forearm has for long been considered a promising framework to allow transradial amputees to control motorized prostheses. In spite of expectations, contemporary muscle–computer interfaces built for this purpose typically fail to satisfy one or more important desiderata, such as accuracy, robustness, and/or naturalness of control, in part due to difficulties in acquiring high-quality signals continuously outside laboratory conditions. In light of such problems, surgically implanted electrodes have been made a viable option that allows for long-term acquisition of intramuscular electromyography (iEMG) measurements of spatially precise origin. As it stands, the question of how information embedded in such signals is best extracted and combined across multiple channels remains open. This study presents and evaluates an approach to this end that uses deep neural networks based on the Long Short-Term Memory (LSTMs) architecture to regress forces exerted by multiple degrees of freedom (DoFs) from multichannel iEMG. Three deep learning models, representing three distinct regression strategies, were evaluated: (I) One-to-One, wherein each DoF is separately estimated by an LSTM model processing a single iEMG channel, (II) All-to-One, wherein each DoF is separately estimated by an LSTM model processing all iEMG channels, and (III) All-to-All, wherein a single LSTM model with access to all iEMG channels estimates all DoFs simultaneously. All models operate on raw iEMG, with no preliminary feature extraction required. When evaluated on a dataset comprising six iEMG channels with concurrent force measurements acquired from 14 subjects, all LSTM strategies were found to significantly outperform a baseline feature-based linear control regression method. This finding indicates that recurrent neural networks can learn to transform raw forearm iEMG signals directly into representations that correlate with forces exerted at the level of the hand to a greater degree than simple features do. Furthermore, the All-to-All and All-to-One strategies were found to exhibit better performance than the One-to-One strategy. This finding suggests that, in spite of the spatially local nature of signals, iEMG from muscles not directly actuating the relevant DoF can provide contextual information that aid in decoding motor intent.

Keywords: iEMG, force, deep learning, LSTM, recurrent neural networks, regression, proportional control, simultaneous control

OPEN ACCESS

Edited by:
Claudio Castellini, University of Erlangen Nuremberg, Germany

Reviewed by:
Andrea Gigli, Helmholtz Association of German Research Centers (HZ), Germany
Matthew Dyson, Newcastle University, United Kingdom

****Correspondence:***
Alexander E. Olsson
alexander.olsson@bme.lth.se
Christian Antfolk
christian.antfolk@bme.lth.se

Specialty section:
This article was submitted to Neuroprosthetics, a section of the journal Frontiers in Neuroscience

Received: *15 September 2021*
Accepted: *26 October 2021*
Published: *17 November 2021*

Citation:
Olsson AE, Malešević N, Björkman A and Antfolk C (2021) End-to-End Estimation of Hand- and Wrist Forces From Raw Intramuscular EMG Signals Using LSTM Networks. Front. Neurosci. 15:777329. doi: 10.3389/fnins.2021.777329

INTRODUCTION

Voluntary movements of fingers are controlled by intrinsic muscles located in the hand and extrinsic muscles, that also control wrist movement, located in the forearm (Blana et al., 2017). After a wrist disarticulation or transradial amputation, whereas the hand itself is lost, the extrinsic muscles, although shortened, will largely remain in place and innervated. Typically, such remnant muscles can still be contracted voluntarily by the amputee and thereby produce detectable myoelectric activity that can be measured with electromyography (EMG). For many decades now (Wirta et al., 1978), processing EMG signals originating from remnant muscles has been considered a leading candidate in the ongoing pursuit of tools that allow amputees to better control prosthetic hands.

The standard form of myoelectric control interface in commercially available upper limb prostheses is known as *direct control* (Paciga et al., 1980). In this framework, pairs of surface electrodes measure the amplitudes of EMG from antagonistic muscle pairs located superficially in the residual limb; the difference between each pair of channels can subsequently be used to control a single degree of freedom (DoF) of an active prosthesis. Although simple and robust, direct control exhibits some crucial disadvantages compared to the functionality afforded effortlessly by the healthy human hand: First, control is not intuitive, as the activation pattern required to perform a movement do not correspond to the physiologically natural pattern. Second, the limited number of electrode pairs that can be accommodated by the approach means that only a handful of DoFs can be controlled simultaneously, hampering dexterity. This is in sharp contrast to the impressive mechanical abilities of contemporary high-end active hand prostheses—such devices could, if provided with sufficiently precise control commands, articulate a large number of DoFs simultaneously and independently (Saikia et al., 2016). This shortcoming of control has been conjectured to be one of the main drivers of the high abandonment rate of myoelectric upper limb prostheses (Biddiss and Chau, 2007).

A somewhat more recent development is myoelectric control based on *pattern recognition* (Hudgins et al., 1993; Scheme and Englehart, 2011; Zecca et al., 2017). In this control framework, the movement intent of the user is inferred by a machine learning model that operates on continuously segmented multichannel surface EMG (sEMG) signals. To learn an appropriate mapping via supervised learning, training data must be provided to the model in the form of example sEMG time windows and corresponding measures of movement intent (e.g., kinematic regressands or categorical target motion classes). Aside from this requirement of initial calibration data, pattern recognition control exhibits many promising advantages compared to the direct control paradigm: the subjective sensation of control can be made completely intuitive, and a relatively large set of DoFs (dependent on the size of the electrode array) can in theory be controlled simultaneously (Scheme and Englehart, 2011). However, due to problems of robustness and stability over time of algorithms, clinical adoption remains uncommon (Jiang et al., 2012). Furthermore, being predicated on sEMG puts practical limits on the kind of information that can be made available to machine learning models of this kind. Signals originating from deeply situated muscles are attenuated to a significant degree, and even superficial muscles can generate levels of crosstalk that hamper the task of separately decoding multiple DoFs (Lowery et al., 2002). Thus, in addition to improving algorithms, a promising avenue for improving control interfaces is to provide algorithms with more informative raw input signals.

By invasively inserting needle- or fine-wire electrodes directly into muscles, EMG signals that have very precise spatial origin can be acquired. Intramuscular EMG (iEMG) signals of this kind exhibit negligible crosstalk and a high degree of correlation with concurrent kinematics (Lowery et al., 2006; Kamavuako et al., 2009), but would most likely be too delicate to use as the basis of control for a wearable system. Surgically implanted electrodes (Brånemark et al., 2001; Hobby et al., 2001; Kuiken et al., 2009) have been proposed as a way to circumvent this problem and are quickly becoming a realistic alternative fit for widespread adoption. With these approaches, individual muscles can be recorded for extensive periods of time, even chronically, thereby potentially providing a long-lived control interface between the user and the prostheses. This kind of setup entails a further benefit of better resisting electrode shift—a phenomenon known to severely impair the performance of pattern recognition control based on sEMG over time due to distributional shifts (Kyranou et al., 2018).

Only a handful of previous studies (Hargrove et al., 2007; Cipriani et al., 2014; Smith et al., 2014, 2016) have experimentally investigated the use of iEMG as the input to prostheses control interfaces, likely in part due to the inherent difficulties in acquiring iEMG signals invasively. Due to the already highly informative content of iEMG signals w.r.t. concurrent kinematics, such studies have justifiably either opted for the use of dual-site, amplitude-proportional linear control (e.g., Smith et al., 2014), or relatively simple pattern recognition algorithms operating on extracted signal features (e.g., Hargrove et al., 2007). Even so, we hypothesize that more sophisticated signal processing could increase the correlation between intramuscular signals and kinematics further, and, in accordance, a more elaborate approach is examined in the current study. Inspired by the success of deep learning for decoding motor intent from sEMG (for reviews of the field, see (Phinyomark and Scheme, 2018; Rim et al., 2020; Buongiorno et al., 2021; Xiong et al., 2021), we use models based on the Long Short-Term Memory (LSTM) architecture (Hochreiter and Schmidhuber, 1997) to regress forces exerted by multiple DoFs at the level of the hand and wrist from multichannel iEMG acquired from extrinsic muscles. Notably, the proposed algorithms evaluated here differ from existing attempts of iEMG force regression in that no feature extraction step is undertaken; rather, the concurrently measured force values are inferred directly, in an end-to-end fashion, from time windows of minimally preprocessed iEMG voltages. Furthermore, to quantify the impact that combining information originating from different muscles has on force regression performance, we let some models operate on a single iEMG channel and some models operate on all channels in order to automatically learn ways to aggregate spatially encoded

information. Whereas the algorithms of the current study were trained and tested offline on a publicly available database, steps were taken to ensure that all parts of the processing pipeline can be executed in a functionally equivalent manner in a real-time scenario.

MATERIALS AND METHODS

Data Acquisition

The database of fine-wire iEMG recordings and concurrent force measurements used in this study was collected for a previous study and has been made publicly available (Malesevic et al., 2020). For the sake of completeness, all properties of the database relevant for the current study are restated in brief here. Data were collected from 14 male, neurologically intact subjects aged between 25 and 57 years (mean age 39 years). All subjects gave informed and written consent prior to participation, and the study was approved by the Regional Ethics Review Board in Lund, Sweden (Dnr 2017-297). Each recording session lasted approximately 30 min and used one out of two possible electrode placement protocols: (I) The Short Residual Limb protocol, targeting the following six muscles of the forearm: flexor carpi radialis (FCR), extensor carpi radialis (ECR), pronator teres (PT), flexor digitorum profundus (FDP), extensor digitorum communis (EDC), and abductor pollicis longus (APL) and (II) The Long Residual Limb protocol, targeting the following six muscles of the forearm: flexor digitorum profundus (FDP), extensor digitorum communis (EDC), abductor pollicis longus (APL), flexor pollicis longus (FPL), extensor pollicis longus (EPL), and extensor indicis proprius (EIP).

Intramuscular EMG (iEMG) signals were sampled at a rate of $F_s = 10240$ Hz. A custom-built measurement device (Malešević et al., 2019) was used to hold the hand stationary for the duration of acquisition (thus ensuring isometric contractions, as would be the case with forearm amputees) and record forces exerted at the level of the hand and wrist. In total, nine force gauges, corresponding to the major degrees of freedom (DoFs) of the hand and wrist, were used: one per finger (D2–D5), two for the thumb, and three for the wrist. Consequently, each session always comprised six channels of iEMG and nine channels of force (synchronized sample-wise). Two of the subjects carried out both protocols, resulting in a dataset comprising 16 recording sessions, out of which eight were recorded with the Short Residual Limb protocol and eight were recorded with the Long Residual Limb protocol. Subjects were assigned to placement protocols randomly.

In total, each recording session comprised 22 unique *tasks*, each corresponding to a movement incorporating either activation of a single or activation of a combination of some or all of the six muscles being assessed in the study. The current study makes use of only the first eight tasks of the database (shown in **Table 1**), representing movements that incorporate a single DoF at a time. Furthermore, to not bias training and test data toward higher force levels and to consequently better simulate the intended use case of prosthesis control, only signals originating from the *sine tracking* substage were used. During this substage, the subject was instructed to contract relevant muscles to track a low-frequency (0.1 Hz) sinusoid with amplitude equal to 20% of the force measured during maximum voluntary contraction plotted in real time on a screen. Each of the eight relevant sine tracking substages comprises 10 periods of such a sinusoid.

TABLE 1 | The subset of movement tasks utilized in the current study.

Code in database	Description
1.X	Index finger: flexion-extension
2.X	Middle finger: flexion-extension
3.X	Ring finger: flexion-extension
4.X	Little finger: flexion-extension
5.X	Thumb: flexion-extension
6.X	Thumb: adduction-abduction
7.X	Wrist: flexion-extension
8.X	Wrist: supination-pronation

Preprocessing

As all algorithms proposed in the current study are intended for online execution, signal preprocessing steps were conducted in a manner compatible with this requirement. To initially reduce the impact of voltage spikes and other outlier samples, all iEMG channels were individually clipped at the 99 and 1 percentile levels and subsequently filtered using a second-order digital Butterworth low-pass filter with a cutoff frequency of 500 Hz. Similarly, all force channels were individually low-pass filtered using a second-order Butterworth filter with a cutoff frequency of 10 Hz. Following these introductory preprocessing steps, all signals were downsampled by a factor of 10 (i.e., to $F_s = 1024$ Hz) in order to reduce computational complexity and facilitate faster convergence of learning algorithms.

Based on iEMG-movement-force matchings provided together with the database, filtered iEMG- and force signals were restructured in the following way: first, each of the nine force channels was separated, on the basis of the sign of the *Tracking Cue* variable provided in the database, into two new channels—one new channel representing positive phase (flexion, adduction, and pronation) and one new channel representing negative phase (extension, abduction, and supination). All 18 resulting phase-specific force channels were subsequently rectified, ensuring that both phases had a positive and similar envelope. Second, each iEMG channel was paired with a single such force channel on the basis of the matchings provided with the database. All force channels that had not been paired with an iEMG channel, i.e., force channels originating from DoFs not actuated by any of the electrode-penetrated muscles, were at this stage discarded. Third, a list of elicited movements (i.e., tasks) that were matched with any of the remaining iEMG-force matchings was created; signal parts originating from elicited movements not contained in the list were discarded and not used further in the current study. As a result, the T time samples remaining following this selection could be represented as two synchronous signal matrices with matched rows: $\mathbf{E} \in \mathbb{R}^{T \times 6}$, containing the six iEMG channels, and $\mathbf{F} \in \mathbb{R}^{T \times 6}$, containing the six matched force channels. Subject-specific elicited movements remaining after this stage are presented in **Table 2**.

In order to represent the resulting data in a way amenable to machine learning methodology, the iEMG signals contained in **E** were segmented into individual regression instances by using a sliding window of width 512 samples (500 ms) with increments of 64 samples (62.5 ms). Each time window was assigned a ground-truth force vector by simply selecting the last row (i.e., time sample) of the sliding window in **F**. In an online application, this would correspond to inferring the current force from the preceding 500 ms of iEMG, with delays between consecutive inferences of 62.5 ms—well in line with acceptable values of myocontrol delay (Farrell and Weir, 2007).

Intramuscular EMG (iEMG) time windows with appertaining force vectors were lastly partitioned into a training set and a test set on the basis of sinusoid period: signal windows originating from the first 7 periods (out of the available 10) of each sine tracking task were designated as training data, signal windows originating from the 8th period were designated validation data, and signal windows originating from the 9th and 10th period were designated as test data. All iEMG windows were linearly rescaled using the channel-wise mean and standard deviation computed from the training set; training iEMG data thus had zero mean and unit variance. All target force values were similarly normalized to have unit variance across training set time windows, but were, due to their rectified nature, not transformed to have non-zero mean.

Deep Learning Models

All deep learning models of the current study were implemented using the TensorFlow 1.12 library (Abadi et al., 2016) and executed in Python 3.6 using a desktop computer equipped with a Nvidia Titan V GPU. All architecture choices and hyperparameters were empirically selected on the basis of performance achieved on the training and validation sets; performance on test set data was never allowed to impact the design of deep learning models.

Three separate strategies for regressing forces, each with a corresponding neural network architecture (all illustrated in **Figure 1**), were implemented in the current study:

1. **One-to-One.** Each individual force channel is estimated by a deep learning model processing a single matching iEMG channel. This approach requires six models in order to infer all output forces—one model per iEMG-force channel pair.
2. **All-to-One.** Each individual force channel is estimated by a deep learning model processing all available iEMG channels. This approach requires six models in order to infer all output forces—one model per force channel to be inferred.
3. **All-to-All.** A single LSTM model operates on all six iEMG channels and estimates all force channels simultaneously. This approach only requires that a single model is trained to infer all output forces.

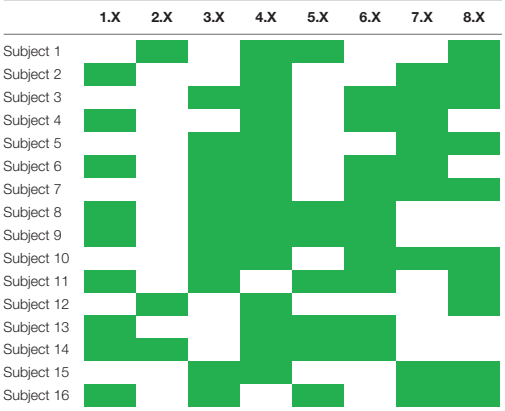
As can be seen in **Figure 1**, the neural model architectures associated with the three above mentioned strategies are almost identical in structure: The input iEMG window is initially filtered by a 1D convolutional layer consisting of 64 filter kernels of size 21 with unit stride. Zero padding was used to keep the time dimension size of the output identical to the time dimension size of the input. Output feature maps of size $512 = 64$ are subsequently fed into the central LSTM layer, whose output (of size 64) at the final time step is fed into two consecutive fully connected layers. All convolutional-, LSTM-, and fully connected layers (except the last) are followed by leaky ReLU activation (Maas et al., 2013), layer normalization (Zhou and Yang, 2019), and dropout with probability 0.2. Notably, only the initial convolutional layer and the final fully connected layer differ in size between regression strategies; thus, the number of trainable parameters (and thus memory footprint and computational complexity) of the different model types (given in **Table 3**) is highly similar.

Irrespective of regression strategy, all deep learning models were fitted to the training data in an identical manner using the AdamW algorithm (Loshchilov and Hutter, 2019) with learning rate $\eta = 10^{-4}$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$ to iteratively minimize the mean squared error loss function. During training, a weight decay of $\lambda = 10^{-9}$ was applied. Training proceeded in minibatches of size 32 until the loss on the validation set had not decreased for 25 consecutive epochs or a total of 250 epochs had passed, whichever came first (i.e., a form of early stopping). For all model types, training was performed independently for each of the 16 recordings in the database. By design, for the One-to-One and All-to-One strategies, one model was trained per output force channel.

Baseline Method

In addition to the deep learning processing pipeline described above, the current study included a conventional, amplitude-proportional linear control method in order to verify that

TABLE 2 | Overview of subject-wise elicited movements selected for regression model training and testing.



Cells shaded green represent elicited movements that remained following preprocessing.

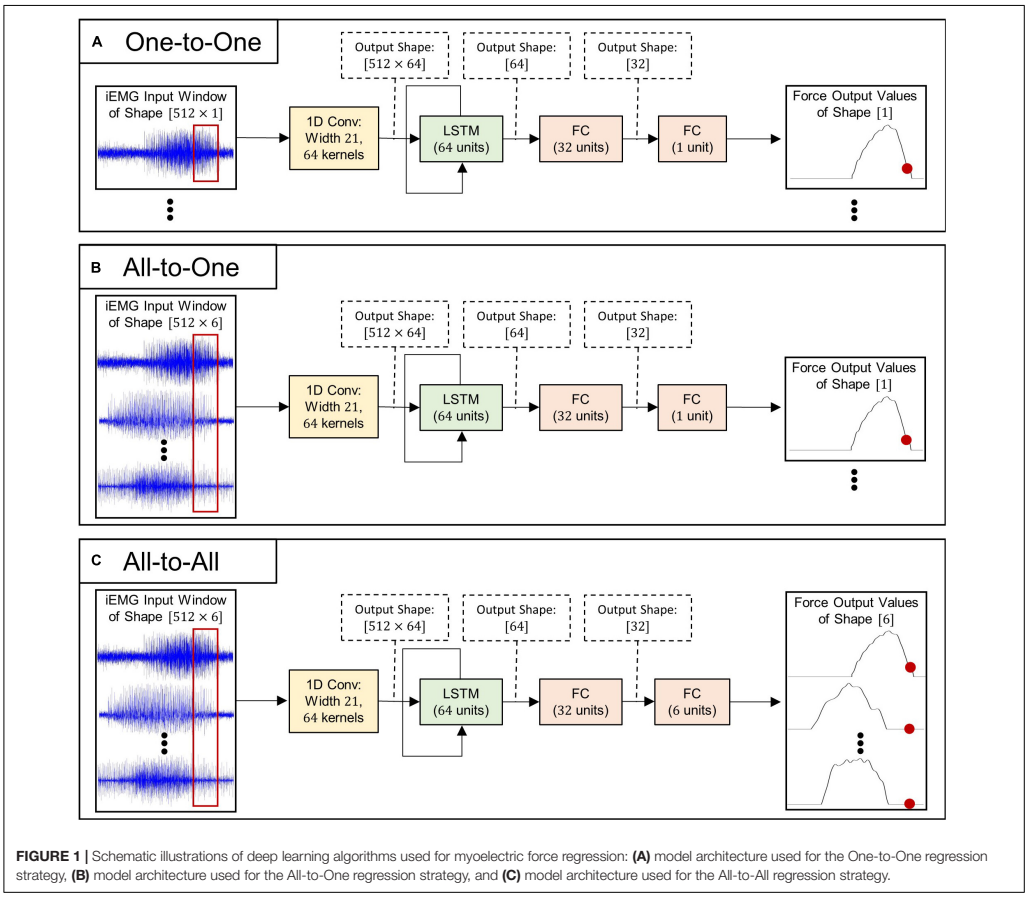


FIGURE 1 | Schematic illustrations of deep learning algorithms used for myoelectric force regression: **(A)** model architecture used for the One-to-One regression strategy, **(B)** model architecture used for the All-to-One regression strategy, and **(C)** model architecture used for the All-to-All regression strategy.

the introduction of more computationally resource-intensive methods was warranted. A simple linear force regression algorithm was selected to represent the status quo for this purpose. The mean absolute values (MAV) feature (Hudgins et al., 1993) was computed channel-wise for each iEMG window in the training set and in the test set. Using the training set data, one univariate ordinary least squares (OLS) linear regression model was fitted to each iEMG channel to predict the concurrent force of its paired force channel. Consequently, this method is here referred to as *One-to-One linear regression*.

Evaluation

The performance of the trained models was evaluated on the test set of iEMG time windows and corresponding ground truth force vectors by computing two standard offline regression metrics: the Root-Mean-Squared Error (RMSE) and the Variance Accounted For (VAF). Each computed scalar value represents the performance of a single regression method on a single recording session from the database.

The RMSE metric quantifies the normalized euclidean distance between the ground truth force values vector and the vector of force values produced by the regression model under

TABLE 3 | Numbers of learnable parameters and inference times of models.

Model	Number of parameters	Wall time of single inference
One-to-One Linear	6 × 2	6 × 1 ms
One-to-One LSTM	6 × 36801	6 × 54 ms
All-to-One LSTM	6 × 43521	6 × 58 ms
All-to-All LSTM	43653	62 ms

consideration. The recording-wise RMSE is here symbolically defined in Equation 1:

$$RMSE = \frac{1}{N}\sum_{n}^{N}\left(\frac{\sqrt{\frac{1}{C}\sum_{c=1}^{C}\left(\left(\hat{y}_{n,c}-y_{n,c}\right)^2\right)}}{y_{max}}\right) \quad (1)$$

where N is the total number of iEMG windows in the test set (variable across recording sessions) and $C = 6$ is the total number of force channels. $y_{n,c}$ and $\hat{y}_{n,c}$ are the ground truth and estimated value, respectively, of the cth force channel associated with the n:th iEMG window and y_{max} is the maximum value of the regressand across the test set. Due to the fact that the RMSE metric increases as the discrepancy between the true and predicted value increases, a lower value represents better regression performance.

The VAF metric quantifies the proportion of variance of the ground truth that the trained model output accounts for. The recording-wise VAF metric is here symbolically defined in Equation 2:

$$VAF = \frac{1}{C}\sum_{c}^{C}\left(1-\frac{Var\left(\hat{\mathbf{y}}_c-\mathbf{y}_c\right)}{Var\left(\mathbf{y}_c\right)}\right) \quad (2)$$

$\mathbf{y}_c$ and $\hat{\mathbf{y}}_c$ are aligned vectors of ground truth and estimated values, respectively, of the cth force channel. A higher VAF metric represents better regression performance.

Statistics

Statistical computations were performed using functions provided with the SciPy library in Python. For all statistical analyses, differences at the $\alpha = 0.05$ level were considered significant. Initially, the non-gaussianity of all metrics and methods was tested with Shapiro–Wilk tests—as the gaussianity of RMSE metrics could not be rejected for any of the regression methods, a one-way repeated measures ANOVA was employed to detect any difference between methods. As a significant difference in RMSE between methods was detected in this way, *post hoc* analyses in the form of paired samples *t*-tests between all pairings of regression methods [$\binom{4}{2} = 6$ comparisons in total] were conducted. Furthermore, the arithmetic mean was selected as the summary statistic to represent the RMSE values achieved by each regression method over all recordings in the database. In contrast, the VAF metric was found to exhibit a significantly non-gaussian behavior for all regression methods. Consequently, a Friedman test was used to establish whether any difference between methods existed. Due to the non-gaussianity of the VAF metric, the median was selected as the summary statistic to represent the VAF values achieved by each regression method over all recordings in the database. As a significant difference in VAF between methods was detected by the Friedman test, *post hoc* analyses in the form of Wilcoxon signed-rank tests between all pairings of regression methods was conducted. For *post hoc* analyses performed on both metrics, acquired *p*-values were subject to Bonferroni correction for multiple comparisons; *p*-values are presented in corrected form throughout the Results section.

RESULTS

Examples of input and output signals produced by all regression methods are shown in **Figure 2**. RMSE and VAF values achieved by all models are summarized graphically in **Figures 3**, **4**, respectively.

For the RMSE metric, one-way repeated measures ANOVA found a significant ($p = 1.4 \cdot 10^{-9}$) difference in performance between regression methods. As per *post hoc* paired samples *t*-tests, the decreases in mean between MAV-based linear regression (mean RMSE 0.123, SD 0.020) and the One-to-One strategy (mean RMSE 0.104, SD 0.018), the All-to-One strategy (mean RMSE 0.081, SD 0.018), and the All-to-All strategy (mean RMSE 0.077, SD 0.016) were 0.019 ($p = 2.9 \cdot 10^{-4}$), 0.042 ($p = 1.1 \cdot 10^{-5}$), and 0.046 ($p = 1.0 \cdot 10^{-6}$), respectively. Thus, all LSTM-based methods exhibited significantly better performance than the baseline method. Furthermore, both the All-to-One strategy and the All-to-All strategy significantly outperformed the One-to-One strategy, with differences in mean of 0.023 ($p = 1.6 \cdot 10^{-6}$) and 0.027 ($p = 2.6 \cdot 10^{-8}$), respectively. Lastly, a non-significant difference in mean of 0.004 ($p = 2.2 \cdot 10^{-1}$) was found between the All-to-One strategy and the All-to-All strategy. Significant differences in RMSE between methods as presented in this section are summarized in **Table 4**.

For the VAF metric, a Friedman test found a significant ($p = 1.9 \cdot 10^{-6}$) difference in performance between regression methods. As per *post hoc* paired samples Wilcoxon signed rank tests, the increases in median between MAV-based linear regression (median VAF 21.5%) and the One-to-One strategy (median VAF 40.3%), the All-to-One strategy (median VAF 59.6%), and the All-to-All strategy (median VAF 60.9%) were 18.8% ($p = 9.0 \cdot 10^{-2}$), 38.0% ($p = 3.8 \cdot 10^{-3}$), and 39.4% ($p = 4.6 \cdot 10^{-3}$), respectively. Thus, all LSTM-based methods with the exception of the One-to-One strategy exhibited significantly better performance than the baseline method. Furthermore, both the All-to-One strategy and the All-to-All strategy significantly outperformed the One-to-One strategy, with differences in median of 19.2% ($p = 5.6 \cdot 10^{-3}$) and 20.6% ($p = 2.6 \cdot 10^{-3}$), respectively. Lastly, a non-significant difference in median of 1.3% ($p = 1.0$) was found between the All-to-One strategy and the All-to-All strategy. Significant differences in VAF between methods as presented in this section are summarized in **Table 5**.

DISCUSSION

The main aim of this study was to investigate the use of deep learning models based on the LSTM architecture to regress forces pertaining to multiple kinematic DoFs from concurrently acquired iEMG—specifically, the possibility of circumventing the need for hand-crafted signal features by letting such models map raw iEMG segments directly to regressand force values. From the

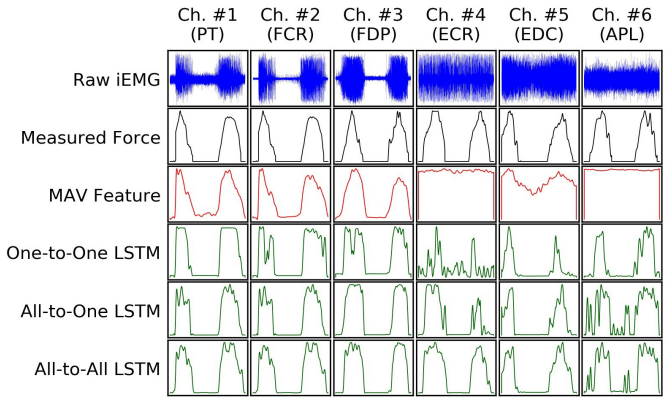


FIGURE 2 | An example (from a random subject) of input iEMG presented to, and output force estimates produced by, all regression models, together with synchronous ground truth force measurements. Segments of iEMG channels (columns) and paired force channels shown here were selected from the task of the recording protocol in which they were maximally correlated; as such, the columns do not represent mutually concurrent time intervals in this illustration.

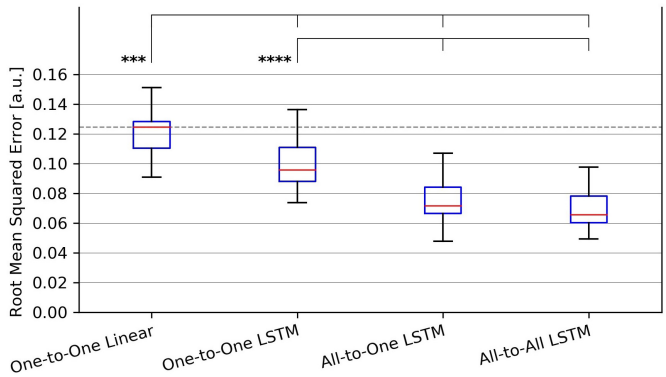


FIGURE 3 | Tukey box plot of Root-Mean-Squared Error (RMSE) metrics achieved by all evaluated force regression methods. Whiskers extend 1.5 interquartile ranges below and above the first and third quartile, respectively. ***$p < 0.001$ and ****$p < 0.0001$.

obtained results, it was apparent that all proposed deep learning regression strategies outperformed the baseline One-to-One linear regression method in the sense of producing significantly lower mean RMSE values across recordings. Furthermore, two out of three deep learning methods were found to produce significantly better VAF metrics than the baseline. Together these findings lend credence to a view of end-to-end force estimation via deep learning methodology in general and via LSTMs in particular as a promising method to increase the accuracy of real-time proportional motor intent decoding through iEMG processing.

An additional aim was to evaluate differences in performance between the three examined deep learning regression strategies: estimating the force exerted by each DoF separately from a single iEMG channel originating from the active muscle (One-to-One), estimating each force channel separately but from all available iEMG channels (All-to-One), or directly estimating all force channels simultaneously from all available iEMG channels (All-to-All). For both computed performance metrics, it was found that the All-to-All and All-to-One strategies significantly outperformed the One-to-One strategy, indicating that information from muscles other than the prime mover increased the accuracy of force estimates. Two explanations of this finding are hypothesized here: First, aside from a single prime mover muscle, multiple synergist muscles can be causally implicated in the actuation of a single DoF. Relatedly, when one

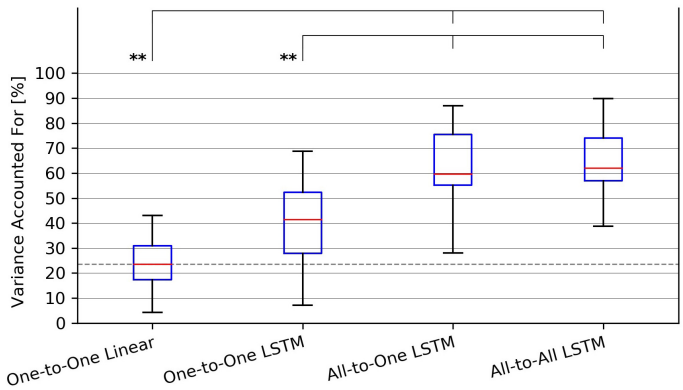


FIGURE 4 | Tukey box plot of Variance Accounted For (VAF) metrics achieved by all evaluated force regression methods. Whiskers extend 1.5 interquartile ranges below and above the first and third quartile, respectively. $^{**}p < 0.01$.

tries to selectively produce isometric force on a specific joint, other joints are usually stabilized by co-contraction, or even opposing contraction of the antagonist muscle. Measurements from such ancillary muscles could thus help regression models paint a more exhaustive picture of the biomechanical state of the arm when estimating forces. Second, even signals from muscles not mechanically involved in the current motion could in theory provide contextual information (of factors such as pose, fatigue, etc.) that correlate non-linearly with exerted force and, by extension, gives rise to better model performance.

TABLE 4 | Paired absolute differences in mean value of the RMSE metric across recording sessions (upper triangular part of table), and corresponding *p*-values (lower triangular part of table), separating regression methods.

	One-to-One Linear	One-to-One LSTM	All-to-One LSTM	All-to-All LSTM
One-to-One Linear		0.019	0.042	0.046
One-to-One LSTM	$p = 2.9 \cdot 10^{-4}$		0.023	0.027
All-to-One LSTM	$p = 1.1 \cdot 10^{-5}$	$p = 1.6 \cdot 10^{-6}$		0.004
All-to-All LSTM	$p = 1.0 \cdot 10^{-6}$	$p = 2.6 \cdot 10^{-8}$	$p = 2.2 \cdot 10^{-1}$	

Pairings exhibiting a significant difference at the $\alpha = 0.05$ level are shaded red.

TABLE 5 | Paired absolute differences in median value of the VAF metric across recording sessions (upper triangular part of table), and corresponding *p*-values (lower triangular part of table), separating regression methods.

	One-to-One Linear	One-to-One LSTM	All-to-One LSTM	All-to-All LSTM
One-to-One Linear		18.8%	38.0%	39.4%
One-to-One LSTM	$p = 9.1 \cdot 10^{-2}$		19.2%	20.6%
All-to-One LSTM	$p = 3.8 \cdot 10^{-3}$	$p = 5.6 \cdot 10^{-3}$		1.3%
All-to-All LSTM	$p = 4.7 \cdot 10^{-3}$	$p = 2.6 \cdot 10^{-3}$	$p = 1.0$	

Pairings exhibiting a significant difference at the $\alpha = 0.05$ level are shaded red.

From the perspective of conserving computational resources, it is promising that the All-to-All strategy performed at a level either higher than or indistinguishable from both of the other deep learning regression strategies; as only a single model is required to regress the forces exerted by all DoFs, both the computational complexity and memory footprint of this strategy are lower compared to the other approaches. This is of particular interest for prosthesis control—algorithms that would need to be implemented in an embedded processing environment, where computational resources are limited. Unfortunately, the resources required by the All-to-All LSTM model are still markedly higher than those of status quo feature-based linear control methods, even when considering the fact that no feature extraction step is necessary in the processing pipeline. The total number of parameters required to instantiate an All-to-All LSTM model was in this study 43,653 (see **Table 3**); assuming single-precision floating-point numbers are used, these require 175 kB to store in memory. Together with all intermittent activation maps (i.e., output volumes of layers) of the model requiring 538 kB to store, this represents a total memory footprint of approximately 713 kB—well within limits of existing embedded processors. As such, it is unlikely that memory footprint would be a limiting factor in realistic scenarios; the main bottleneck in this regard is instead likely the inference delay of the model. In principle, the highest acceptable inference delay is equal to the time between consecutive iEMG windows—with this delay, force values of windows are inferred at the same rate as that which they are sampled at. Whereas the inference delay of the All-to-All strategy (**Table 3**) narrowly falls below the window step size of 62.5 ms, values were measured from a model running on a GPU-equipped desktop computer. In an embedded application, the window duration could be decreased, and/or the window separation time could be increased, to reduce the real-time computational burden. However, for control delay not to become noticeable by the prosthesis user, the delay of the control system

should fall below 300 ms (Englehart and Hudgins, 2003). With mechanical delays inherent to the prosthesis itself, this leaves approximately 125 ms available for algorithm-induced delays (Farrell and Weir, 2007). To circumvent these issues, a possibility is to let LSTM models operate continuously on iEMG samples as they are acquired instead of on a window-wise basis, as has been considered in previous studies using sEMG (Olsson et al., 2019). Nevertheless, it is apparent that future work that focuses on finding more computationally efficient LSTM regression architectures would be of value. As all algorithms considered in the current study were both trained and evaluated on a publicly available dataset, comparisons with alternative methods can be carried out transparently and straightforwardly.

Aside from the questions of computational complexity discussed above, a salient limitation of the approach taken in the current study is the necessity of regressand force values. Naturally, for amputee prosthesis users, acquisition of target force values prior to model training would not be possible. For the supervised learning approach to model training taken in the current study to be made applicable, some appropriate proxy regressand would thus have to substitute for forces measured at the level of the hand and wrist. A well-tried candidate solution is the use of mirrored training (Nielsen et al., 2011), whereby force measurements would be taken from the intact, contralateral hand while the amputee performs motions bilaterally. Another possibility is to ask the user to slowly increase and decrease the intensity of muscle contraction in accordance with some visual cue and subsequently use said cue as ground truth to be inferred from EMG, as has been investigated previously (Ameri et al., 2019).

The superior performance of deep learning methods in the current study is in line with findings from the general machine learning literature that indicate that signal representations automatically learned from data are oftentimes more informative than features designed manually toward the same end (Bengio et al., 2013). An intriguing research direction of specific interest for developing embedded, online motor decoding systems is that of reverse engineering the content of learned EMG features. If carried out successfully, this project could allow classical methods to enjoy some of the higher performance exhibited by deep learning methods without as high computational costs. Furthermore, for the purpose of leveraging the finding that information from multiple muscles improve performance, comparisons with sensor fusion techniques that allow for non-linear feature interactions (e.g., kernel regression; Bishop, 2006), but are less computationally demanding than recurrent neural networks, could be the focus of future work.

DATA AVAILABILITY STATEMENT

Publicly available datasets were analyzed in this study. These data can be found here: https://figshare.com/s/06f113bd74ecf6384729.

ETHICS STATEMENT

The studies involving human participants were reviewed and approved by the Regional Ethics Review Board in Lund, Sweden. The patients/participants provided their written informed consent to participate in this study.

AUTHOR CONTRIBUTIONS

AO implemented the deep learning algorithms, analyzed the results, created the figures, and drafted the manuscript. NM, AB, and CA provided critical comments and finalized the manuscript for publication. All authors contributed to the design of the study.

FUNDING

This research was supported by the Promobilia Foundation, the Crafoord Foundation, the European Commission under the DeTOP project (LEIT-ICT-24- 2015 and GA #687905), and the Swedish Research Council (DNR 2019-05601).

ACKNOWLEDGMENTS

We gratefully acknowledge the support of the NVIDIA Corporation with the donation of the Titan V GPU used for this research.

REFERENCES

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., et al. (2016). "TensorFlow: a system for large-scale machine learning," in *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation, OSDI*, (Berkeley, CA: USENIX Association).

Ameri, A., Akhaee, M. A., Scheme, E., and Englehart, K. (2019). Regression convolutional neural network for improved simultaneous EMG control. *J. Neural Eng.* 16:036015. doi: 10.1088/1741-2552/ab0e2e

Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: a review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* 35, 1798–1828.

Biddiss, E., and Chau, T. (2007). Upper-limb prosthetics: critical factors in device abandonment. *Am. J. Phys. Med. Rehabil.* 86, 977–987. doi: 10.1097/phm.0b013e3181587f6c

Bishop, C. M. (2006). "Kernel methods," in *Pattern Recognition and Machine Learning*, ed. H. Van Dyke Parunak (Berlin: Springer-Verlag), 291–320.

Blana, D., Chadwick, E. K., van den Bogert, A. J., and Murray, W. M. (2017). Real-time simulation of hand motion for prosthesis control. *Comput. Methods Biomech. Biomed. Engin.* 20, 540–549.

Brånemark, R., Brånemark, P. I., Rydevik, B., and Myers, R. R. (2001). Osseointegration in skeletal reconstruction and rehabilitation: a review. *J. Rehabil. Res. Dev.* 38, 175–181.

Buongiorno, D., Cascarano, G. D., De Feudis, I., Brunetti, A., Carnimeo, L., Dimauro, G., et al. (2021). Deep learning for processing electromyographic signals: a taxonomy-based survey. *Neurocomputing* 452, 549–565. doi: 10.1016/j.neucom.2020.06.139

Cipriani, C., Segil, J. L., Birdwell, J. A., and Weir, R. F. (2014). Dexterous control of a prosthetic hand using fine-wire intramuscular electrodes in targeted extrinsic

muscles. *IEEE Trans. Neural Syst. Rehabil. Eng.* 22, 828–836. doi: 10.1109/TNSRE.2014.2301234

Englehart, K., and Hudgins, B. (2003). A robust, real-time control scheme for multifunction myoelectric control. *IEEE Trans. Biomed. Eng.* 50, 848–854. doi: 10.1109/TBME.2003.813539

Farrell, T. R., and Weir, R. F. (2007). The optimal controller delay for myoelectric prostheses. *IEEE Trans. Neural Syst. Rehabil. Eng.* 15, 111–118. doi: 10.1109/tnsre.2007.891391

Hargrove, L. J., Englehart, K., and Hudgins, B. (2007). A comparison of surface and intramuscular myoelectric signal classification. *IEEE Trans. Biomed. Eng.* 54, 847–853. doi: 10.1109/tbme.2006.889192

Hobby, J., Taylor, P. N., and Esnouf, J. (2001). Restoration of tetraplegic hand function by use of the neurocontrol freehand system. *J. Hand Surg. Am.* 26, 459–464. doi: 10.1054/jhsb.2001.0587

Hochreiter, S., and Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.* 9, 1735–1780.

Hudgins, B., Parker, P., and Scott, R. N. (1993). A new strategy for multifunction myoelectric control. *IEEE Trans. Biomed. Eng.* 40, 82–94.

Jiang, N., Dosen, S., Muller, K. R., and Farina, D. (2012). Myoelectric control of artificial limbs: is there a need to change focus? [in the spotlight]. *IEEE Signal Process. Mag.* 29, 150–152.

Kamavuako, E. N., Farina, D., Yoshida, K., and Jensen, W. (2009). Relationship between grasping force and features of single-channel intramuscular EMG signals. *J. Neurosci. Methods* 185, 143–150. doi: 10.1016/j.jneumeth.2009.09.006

Kuiken, T. A., Li, G., Lock, B. A., Lipschutz, R. D., Miller, L. A., Stubblefield, K. A., et al. (2009). Targeted muscle reinnervation for real-time myoelectric control of multifunction artificial arms. *JAMA* 301, 619–628. doi: 10.1001/jama.2009.116

Kyranou, I., Vijayakumar, S., and Erden, M. S. (2018). Causes of performance degradation in non-invasive electromyographic pattern recognition in upper limb prostheses. *Front. Neurorobot.* 12:58. doi: 10.3389/fnbot.2018.00058

Loshchilov, I., and Hutter, F. (2019). "Decoupled weight decay regularization," in *Proceedings of the 7th International Conference on Learning Representations*, (Lisbon: International Conference on Learning Representations).

Lowery, M. M., Stoykov, N. S., Taflove, A., and Kuiken, T. A. (2002). A multiple-layer finite-element model of the surface EMG signal IEEE Trans. *Biomed. Eng.* 49, 446–454. doi: 10.1109/10.995683

Lowery, M. M., Weir, R. F. F., and Kuiken, T. A. (2006). Simulation of intramuscular EMG signals detected using implantable myoelectric sensors (IMES). *IEEE Trans. Biomed. Eng.* 53, 1926–1933. doi: 10.1109/TBME.2006.881774

Maas, A. L., Hannun, A. Y., and Ng, A. Y. (2013). "Rectifier nonlinearities improve neural network acoustic models," in *Proceedings of the ICML Workshop on Deep Learning for Audio, Speech and Language*, (Atlanta: JMLR W&CP).

Malešević, N., Andersson, G., Björkman, A., Controzzi, M., Cipriani, C., and Antfolk, C. (2019). Instrumented platform for assessment of isometric hand muscles contractions. *Meas. Sci. Technol* 30:065701. doi: 10.1088/1361-6501/ab0eae

Malesevic, N., Björkman, A., Andersson, G. S., Matran-Fernandez, A., Citi, L., Cipriani, C., et al. (2020). A database of multi-channel intramuscular electromyogram signals during isometric hand muscles contractions. *Sci. Data* 7:10. doi: 10.1038/s41597-019-0335-8

Nielsen, J. L. G., Holmgaard, S., Jiang, N., Englehart, K. B., Farina, D., and Parker, P. A. (2011). Simultaneous and proportional force estimation for multifunction myoelectric prostheses using mirrored bilateral training. *IEEE Trans. Biomed. Eng.* 58, 681–688. doi: 10.1109/TBME.2010.2068298

Olsson, A., Malesevic, N., Bjorkman, A., and Antfolk, C. (2019). "Exploiting the intertemporal structure of the upper-limb semg: comparisons between an LSTM network and cross-sectional myoelectric pattern recognition methods," in *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, (Berlin: Institute of Electrical and Electronics Engineers). doi: 10.1109/EMBC.2019.8856648

Paciga, J. E., Richard, P. D., and Scott, R. N. (1980). Error rate in five-state myoelectric control systems. *Med. Biol. Eng. Comput.* 18, 287–290. doi: 10.1007/BF02443381

Phinyomark, A., and Scheme, E. (2018). EMG pattern recognition in the era of big data and deep learning. *Big Data Cogn. Comput.* 2:21.

Rim, B., Sung, N. J., Min, S., and Hong, M. (2020). Deep learning in physiological signal data: a survey. *Sensors* 20:969. doi: 10.3390/s20040969

Saikia, A., Mazumdar, S., Sahai, N., Paul, S., Bhatia, D., Verma, S., et al. (2016). Recent advancements in prosthetic hand technology. *J. Med. Eng. Technol.* 40, 255–264.

Scheme, E., and Englehart, K. (2011). Electromyogram pattern recognition for control of powered upper-limb prostheses: state of the art and challenges for clinical use. *J. Rehabil. Res. Dev.* 48:643. doi: 10.1682/jrrd.2010.09.0177

Smith, L. H., Kuiken, T. A., and Hargrove, L. J. (2014). Real-time simultaneous and proportional myoelectric control using intramuscular EMG. *J. Neural Eng.* 11:066013. doi: 10.1088/1741-2560/11/6/066013

Smith, L. H., Kuiken, T. A., and Hargrove, L. J. (2016). Evaluation of linear regression simultaneous myoelectric control using intramuscular EMG. *IEEE Trans. Biomed. Eng.* 63, 737–746.

Wirta, R. W., Taylor, D. R., and Finley, F. R. (1978). Pattern-recognition arm prosthesis: a historical perspective. A final report. *Bull. Prosthet. Res* 12, 8–35.

Xiong, D., Zhang, D., Zhao, X., and Zhao, Y. (2021). Deep learning for EMG-based human-machine interaction: a review. *IEEE/CAA J. Autom. Sin.* 8, 512–533. doi: 10.1109/jas.2021.1003865

Zecca, M., Micera, S., Carrozza, M. C., and Dario, P. (2017). Control of multifunctional prosthetic hands by processing the electromyographic. *Signal Crit. Rev. Biomed. Eng.* 30, 459–485.

Zhou, X. Y., and Yang, G. Z. (2019). Normalization in training U-Net for 2-D biomedical semantic segmentation. *IEEE Robot. Autom. Lett.* 11:14105. doi: 10.1109/lra.2019.2896518

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

Paper VI

Calibration-free myoelectric decoding

A. E. Olsson, N. Malešević, A. Björkman, C. Antfolk

Manuscript

Calibration-Free Myoelectric Decoding

Alexander E. Olsson, Nebojša Maleševic, Anders Björkman, Christian Antfolk

Abstract

Machine learning (ML) has for a long time been the central tool for *motor-intent decoding* of surface electromyography (sEMG) signals, with envisioned uses of ML-based muscle-computer interfaces ranging from prosthetics to consumer electronics. Existing myoelectric pattern recognition algorithms, albeit functional, require the collection of substantial amounts of gesture- and user-specific training data to achieve acceptable performance, making them impractical for many purposes. This paper presents a ML-based framework for efferent myoelectric interfacing intended to work for new users without collecting *any* new training data. At the core of our approach lies a *geometry-aware* Transformer-based model architecture that (i) can operate on input EMG windows collected from arbitrary electrode configurations and (i) outputs a generalized intent representation. These properties make it possible to train a single model instantiation on virtually all publicly available EMG databases. We initially pretrain our model on a selection of 30 such databases, together comprising 510 subjects and 108 unique gestures. The resulting configuration-agnostic model was thereafter both evaluated directly and finetuned on the NinaPro DB2 and DB3 databases and evaluated w.r.t. inter-subject performance. We demonstrate that user-agnostic models finetuned on Ninapro DB2 are competitive with user-specific Linear Discriminant Analysis (LDA) models, indicating the feasibility of calibration-free intent decoding. However, as trained models are much larger than what would fit on embedded hardware, modifications are needed to make the method practically viable. Furthermore, all interuser models of this study underperform the user-specific LDA status quo on amputee data, indicating complementary approaches might be necessary for prosthetic applications.

1 Introduction

The surface electromyogram (sEMG) recorded from muscles during voluntary movement represents the aggregated electrical activity of the individual motor units that contribute to the execution of the movement[1]. Due to its non-invasive nature and the direct relationship with the latent neural drive, sEMG has many times been identified as a practical path towards efferent neural interfacing[2]. As a control modality, sEMG has been applied generously by researchers for purposes such as virtual reality[3], rehabilitation robotics[4], and, perhaps most saliently, upper-limb prosthetics[5]. However, despite satisfying several practical desiderata and showing promise in an abundance of studies, intuitive sEMG-driven control has yet to make a broad impact in human-computer interfacing, clinical or otherwise, outside the lab[6].

One potential explanation for the apparently unrealized potential of myoelectric interfaces stems from the fact that the relationship between sEMG and concurrent kinematics is highly nonlinear and characterized by apparent stochasticity. In the field of upper-limb prosthetics, which places stringent demands on control system robustness, this issue has typically been managed by resorting to using *direct control*[7]. This approach, wherein a limited number of

kinematic degrees of freedom (DoFs) are controlled linearly by sEMG amplitude collected from antagonistic muscle pairs, is not amenable to intuitive control over multiple kinematic Degrees of Freedom (DoFs), limiting its usefulness in, for example, approximating the full dexterity of the human hand for the purpose of controlling multifunctional prostheses or interacting with virtual environments.

To create interfaces that allow for natural and high-throughput extraction of movement information, *machine learning* (ML) (or, interchangeably, *pattern recognition*) has for a long time been viewed as the most, if not the only, viable approach. In this paradigm, the problem of modelling the complicated mapping from sEMG to (some measurable proxy of) movement intent is circumvented by fitting a statistical model to example pairings of sEMG windows and either a categorical (e.g. gesture) or or continuous (e.g. joint kinematics) target, making the problem one of either classification or regression, respectively. To reduce the complexity of the task, informative and motion-discriminative features are typically extracted and used as the input to myoelectric motor decoding systems[8]. More recently, studies on large Artificial Neural Network (ANN) models have indicated that feature engineering might be considered superfluous if sufficient data is available to learn signal representations 'from scratch'[9]. ML-based decoding systems, whether they are operating on features or raw sEMG, have many times demonstrated impressive classification accuracies for wide ranges of performable motions. Even so, due to large variation in movement-conditional sEMG between users individuals[10], a considerable amount of user-specific training data is needed to achieve an acceptable level of performance. Conjecturally, this need to tailor models to individual users has been a significant obstacle to clinical applications, and downright prohibitive for commercial and industrial applications[11].

A number of techniques has been proposed to ease the training burden of myoelectric decoding systems, so far primarily in the context of prosthetics. Matsubara and Morimoto[12] reduced the error rate of a cross-user classifier by identifying components of feature vectors associated with interuser variation. Khushaba *et al.*[13] employed canonical correlation analysis (CCA) to map feature vectors from individual users to those of an expert. More recently, a number of cross-subject approaches, such as transfer learning[14] and adaptive domain-adversarial training[15], have been explored for deep neural networks. While the full impact of such innovations have yet to be determined, they all require the collection of some non-negligible amount of data from the actual end user.

In image- and natural language processing, it has been found that pretraining large models on related objectives often gives rise to surprisingly capable models. Famous examples of so-called *weakly supervised* or *self-supervised* pretraining include Mahajan *et al.*[16], who found that initializing a Convolutional Neural Network (CNN) by predicting hashtags associated with an Instagram dataset improved downstream image classification, and the GPT family of models[17, 18, 19], which have achieved remarkable success on a number of language processing tasks simply by optimizing for next token prediction over a large text corpus.

In the domain of sEMG data, no vast reservoirs of unlabelled signals yet exists. However, a number of open dataset have been introduced comprising hundreds of users and unique gestures. To the best of our knowledge, no previous work has attempted to aggregate more than one dataset for the training of a single mode, as different dataset are typically obtained from varied and idisynchratic electrode geometries . The work presented in this paper is an attempt at circumventing such issues and incorporating multiple datasets in the training of a single, unified model. The central part of our model is a Transformer[20] model that can operate on arbitrary measurement geometries and produce arbitrary intent representations. We initially train this model on a selection of 30 sEMG databases and use the NinaPro DB2

and DB3 databases[21] for finetuning and testing. The results demonstrate the potential of our approach to calibration-free intent decoding, as we show user-agnostic models exhibit similar performance when compared to user-calibrated benchmark models.

2 Experiments

2.1 Data

A full list of the 30 databases used for pretraining in this study is presented in Table 1.

The raw digital sEMG signals of the databases were all initially resampled to the same temporal resolution of $F_s = 1$ kHz using the standard fractional resampling approach[22] in a channel-wise manner. Databases with a native sampling rate of over 1 kHz were initially bandpass filtered (using an 8th order Butterworth filter) with a passband from 10 to 450 Hz and thereafter downsampled. Conversely, databases with a native sampling rate under 1 kHz were appropriately upsampled and bandpass filtered to conunteract aliasing.

A sliding window of size 200 ms and an overlap of 50 ms was used to segment all databases into discrete sEMG windows. As it proved infeasible to compute representative channel-wise summary statistics across all databases, we instead employed a window-wise normalization strategy similar to those of [23] and [24]. Each sample window was linearly rescaled to the range $[-1, 1]$, where 1 represents the maximum (absolute) voltage observed in the window. While this approach destroys all information on absolute amplitude, it maintains both the relative amplitudes of channels and their respective spectral morphologies.

While it would be possible to represent the ground-truth target movement of each window using one-hot encoding over all 108 unique gestures and grasps, this approach would arguably come with a problem. Many movement are only present in a few or even a single dataset, making them difficult to recognize in arbitrary measurement geometries. For this reason, we instead employ a multi-label intent encoding strategy highly similar to that introduced in [24]. We define 18 binary basis labels (named and listed in Table 2) corresponding to the major degrees of freedom (DoFs) of the hand and wrist. Each of the 108 movements was decomposed (using a manually constructed lookup table) to this representation and stored together with relevant sEMG windows. Notably, all windows where no labels at all were present (i.e. the rest class) were excluded. This aided significantly in both easing the computational burden of training and in the design of the ANN model (see section 2.2). As detection and separation of rest is a relatively simple problem that could be solved in practice by a separate, simpler activity detection model, this was deemed a reasonable choice. After the exclusion of sEMG windows coinciding with rest, a total of 9.2 million sEMG windows remained and were stored as a unified pretraining dataset.

As is discussed in the next section, the model introduced here operates not only in sEMG time windows, but furthermore require the specification of the positions of the channels comprising the measurement geometry. Unfortunately, none of the public databases used in this study provide quantitative data on the absolute or relative positions of electrodes during signal acquisition. However, as all provide either photographs of the measurement setup, qualitative descriptions thereof, or both, an approximate lookup table could be constructed in order to map individual channels to corresponding locations for all databases. Concretely, the distance of each electrode from the elbow and its angle from the top of the forearm was manually estimated, from which the full 3D coordinates could be computed assuming a simplified model of the forearm as a cylinder.

The exercise B sub-databases of Ninapro DB2 and DB3 datasets were in this study used both for (i) finetuning models to a specific geometry and (ii) for quantifying performance. The sub-database comprises 17 non-rest movement (Table 3), each repeated 6 times 5 seconds. Although 12 channels of sEMG ($F_s = 2$ kHz) are provided, we discarded channels 11 and 12 (corresponding to the Biceps Brachii and Triceps Brachii, respectively) as none of the pretraining databases contain electrodes placed outside the forearm. For both DB2 and DB3, preprocessing was performed similarly to the pretraining database: sEMG signals were bandpass filtered at 10 to 450 Hz, decimated to a sampling rate of $F_s = 1$ kHz and segmented into 200 ms windows with 50, and each window was annotated in accordance with the movement class with which it coincided (using the postprocessed movement stimuli present in the database). To enable zero-shot inference via the principles in outlined in section 2.3.1, a table containing the decomposition of the movement classes into corresponding multi-label representations was created (see Table 3). Like previously, sEMG windows coinciding with rest were discarded.

2.2 Geometry-Aware Motion Recognition Model

A graphical overview of the model architecture is depicted in Figure 1. The model takes two inputs, the first of which is a 200 samples long sEMG windows (at $F_s = 1$ kHz, i.e. 200 ms) consisting of N signal channels (as stated previously, N may vary between inputs). The second input is a $N \times 3$ *position matrix* $\boldsymbol{P}$, where $\boldsymbol{P}_{n,1}$, $\boldsymbol{P}_{n,2}$, and $\boldsymbol{P}_{n,3}$ represents the x-, y-, and z-coordinate, respectively, of the nth channel. The coordinates are specified in the following manner: In a coordinate system the forearm is modelled as a cylinder with radius and length 1, the x-coordinate goes from palmar to dorsal, and y-coordinates goes from medial to lateral. The z-axis represents the distal direction, where $z = 0$ is the elbow and $z = 1$ the wrist.

The sEMG window is processed channel-wise by a *content encoder* network, The content encoder network is a convolutional neural network (CNN)[25] whose parameters are shared between input channels. It consists of 24 consecutive convolutional layers, each of width 26, with GELU activation functions[26], all followed by layer normalization[27]. Zero-padding (asymmetric, from the left) is used before every layer to keep the length (200) of the output volumes constant through the network. The number of kernels differs between layers: 64 for layers 1 to 6, 128 for layers 7 to 12, 256 for layers 13 to 18, and 512 for layers . All layers make use of residual connections[28]; for layers where the input and output has differing channel number (layers 1, 7, and 19), the method from [28] is used. After the last convolutional layer, the size of the output volume is $N \times 200 \times 512$; its values at the final time instant of the window (i.e. a $N \times 512$ sequence of feature vectors) are fed channel-wise trough a linear, fully connected layer to produce a $N \times 1024$ sequence of content encodings denoted $\boldsymbol{CE}$.

Channel-wise content encodings are *positionally encoded* in order to (i) supply the model with information on the underlying measurement geometry, and (ii) supplement the permutation-invariance of transformer models. In the model presented here, this positional encoding is performed on the basis of channel coordinates. For each channel n, the coordinate vector $[\boldsymbol{P}_{n,1}, \boldsymbol{P}_{n,2}, \boldsymbol{P}_{n,3}]^T$ is fed through a *position encoding network*, consisting of an MLP with 4096 hidden neurons and 1024 outputs applied channel-wise, producing a $N \times 1024$ sequence of positional encodings $\boldsymbol{PE}$. The combined signal representation $\boldsymbol{X}$ to be processed further is then obtained simply from element-wise addition as $\boldsymbol{X} = \boldsymbol{CE} + \boldsymbol{PE}$.

Like the vision transformer[29] (in turn inspired by BERT[30]), the sequence of channel encodings to be fed through the transformer is prepended with an additional learnable class encoding $\boldsymbol{x}_{class}$ of size 1024. The input $\boldsymbol{Z} = [\boldsymbol{x}_{class}, \boldsymbol{X}]^T$ to the transformer is thus a $(N + 1) \times 1024$ sequence.

The transformer is of the *encoder-only* type, identical in structure to the vision transformer. It comprises 24 layers, each composed of (i) multi-head self-attention (MSA) module with 16 heads (each with a key, value, and query size of 64), followed by (ii) a point-wise multi-layer perceptron (MLP) module with a hidden size of 4096 using a GELU activation function after the hidden layer. Like in most Transformer implementations, both the MSA and MLP module make use of layer normalization and residual connections. The output $\boldsymbol{O}$ of the transformer is a sequence of shape $(N+1) \times 1024$; the first element $\boldsymbol{O}_{1,*}$ in the sequence (corresponding to the learnable class encoding) is considered a fixed-length representation of the EMG window, encoding both the content of the EMG signals and the geometry used for acquiring them.

The output representation is either fed to a classifier head or a multi-label head, depending on training objective.

The multi-label head is used during pretraining. It is a two-layer MLP with 4096 hidden neurons and 18 outputs $\hat{\boldsymbol{y}}$ (one per movement label). Although the purpose of this layer is to perform multi-label classification, the standard single-label softmax activation function is applied to the output. Similar to [31] and [16], we found softmax outputs more conducive to optimization than standard label-wise sigmoid. The details of how the output of the multi-label head is handled during training and inference are given in sections 2.3 and 2.3.1, respectively.

When finetuning on a dataset with static measurement geometry (Ninapro DB2 or DB3 in this study), the transformer output representation is fed to the classifier head. The classifier head is composed of a single fully connected layer with softmax that maps $\boldsymbol{O}_{1,*}$ to 17 outputs $\hat{\boldsymbol{c}}$, each representing the (approximate) probability of the input belonging to one of the 17 movement classes in Exercise B of Ninapro DB2 and DB3 (see Table 3).

Counting the position-encoding MLP, the CNN, the linear projection into content encodings, the Transformer, and both output heads, the model has 315 million free parameters (weights and biases) in total. The model can handle arbitrary number of channels N (up to memory constraints, which due to the transformer module grows like $\mathcal{O}(N^2)$).

2.3 Pretraining

During pretraining, a single instantiation of the model described in the previous section (with a multi-label head) was fitted to the data from Table 1, comprising 9.2 million examples. Each example took the form of a tuple $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{P})$, where $\boldsymbol{x} \in \mathbb{R}^{N \times 200}$ is a 200 ms long N-channel sEMG window, $\boldsymbol{y} \in \{0,1\}^{18}$ the DoF-wise label representation of the concurrent movement, and $\boldsymbol{P} \in [-1,1]^3$ the 3D coordinates of the N electrodes.

Categorical cross-entropy was used as the loss function to be minimized. As the ground truth target labels $\boldsymbol{y}$ do not sum to 1 (they do not technically describe a probability distribution), they were linearly renormalized. While no theoretical justification for this step exists, we, like previous work[31, 16], found that this loss formulation facilitates convergence significantly better in multi-label settings than the seemingly more reasonable label-wise sigmoid approach[24]. With $\hat{\boldsymbol{y}}$ as the output of the multi-label head, the loss associated with a single example thus be formulated as:

$$\mathcal{L}_{pretrain} = \sum_{j=1}^{18} D_{KL}\left(\frac{y_i}{\sum_{k=1}^{18} y_k} || \hat{y}_i\right) \text{ where } D_{KL}(p||q) = p \cdot \log \frac{p}{q} \quad (1)$$

$$\text{and } \log \frac{0}{0} \coloneqq 0$$

The batch-wise loss was thereafter obtained as the uniformly weighted average over the examples comprising the batch.

The Adam algorithm[32] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a batch size of 4096 was used to minimize $\mathcal{L}_{pretrain}$ with respect to the parameters via backpropagation. All parameters were initialized using Glorot intialization[33]. Training proceeded for 100000 iterations, using a base learning rate of $\eta_{base} =$ with linear warmup for $S_{warmup} = 10000$ iterations and inverse square root decay, such that the learning rate η_i at iteration i can be described by:

$$\eta_i = \eta_{base} \cdot \min\left(\frac{i}{S_{warmup}}\ ,\ \sqrt{\frac{S_{warmup}}{i}}\right) \tag{2}$$

To not bias training towards databases with long recordings, a special sampling scheme was used to construct individual training batches. Each of the 30 databases were assign a simple quality metric:

$$Q_i = S_i \cdot \sqrt{M_i \cdot C_i} \tag{3}$$

Where S_i, M_i, and C_i are the number of subjects, unique movements, and sEMG channels, respectively, of the ith database. Batches were constructed by random sampling from each database weighted by its respective quality metric. As such, the number of elapsed epochs at the end of training (100000 iterations with batch size 4096) was not constant across databases. For example, 19.1 epochs elapsed for the Hyser database[34], whereas only 1.1*epochs* elapsed for the database from Khusbaba *et al*[35].

Some further modifications was performed to individual examples before being fed to the model. To limit memory usage during training (by far the most significant bottleneck in this study), each sEMG window was limited to only 16 channels. For examples originating from databases with larger electrode arrays, 16 channels were randomly selected at the sampling of each window. For databases with less than 16 channels, all channels were included and the input zero-padded to conform to the standard batch tensor size. Furthermore, the input position matrix $\boldsymbol{P}$ was corrupted with additive, white Gaussian noise with mean 0 and standard deviation 0.2 to represent the uncertainty in the manually estimated positions of the training data

Dropout[36] with probability 10% and label smoothing[37] of 0.1 was employed during training. Dropout was applied following every convolutional layer, MSA module, and after the hidden layer of all MLPs.

2.3.1 Zero-shot Inference

As described, the pretrained ANN model with a multi-label head outputs a probability distribution $\hat{y} \in (0, 1)^18$ over movement labels. Naturally, this output representation lacks inherent meaning—by construction, more than one label can be associated with each input sEMG window and position matrix. Thus, in order to be able to use the pretrained model for inference, we propose two simple algorithms for converting $\hat{y}$ into a label set $\mathbf{l} \in \{0, 1\}^{18}$, representing (estimate of) the true discrete movement labels coinciding with the input sEMG window.

The first algorithm (Algorithm 1) works by iteratively constructing the label set by adding labels in decreasing order of probability. The process stops when the total probability mass of all previously selected labels exceeds a threshold value $T \in [0, 1)$ (a hyperparameter).

Algorithm 1 Threshold-based Label Detection

Require: $\hat{\boldsymbol{y}}$: probability distribution over labels, T: hyperparameter threshold value
Ensure: $\hat{\boldsymbol{l}}$: binary vector representing detected labels
Initialize $\hat{\boldsymbol{l}} \leftarrow \mathbf{0}$ of size 18
Initialize $currentSum \leftarrow 0$
while $currentSum < T$ **do**
    Find index j such that $\hat{y}_j = \max(\hat{\boldsymbol{y}})$
    Set $\hat{l}_j \leftarrow 1$
    Update $currentSum \leftarrow currentSum + \hat{y}_j$
    Set $\hat{y}_j \leftarrow -\infty$ to exclude it from future consideration
end while

The second algorithm (Algorithm 2) works by comparing the output distribution to a set of externally provided candidate label sets. The candidate label sets are converted to candidate probability distributions via linear renormalization, as was done to the ground truth targets during pretraining. The detected label set is then set to equal the label set whose renormalized distribution is the most similar (as measured by KL-divergence) to that produced by the ANN model.

Algorithm 2 Label Detection Based on Candidate Similarity

Require: $\hat{\boldsymbol{y}}$: input probability distribution, $\mathcal{C}$: list of candidate label sets
Ensure: $\hat{\boldsymbol{l}}$: binary vector representing detected labels
Initialize $\mathcal{D} \leftarrow$ empty list
for each $c \in \mathcal{C}$ **do**
    Renormalize c to create a probability distribution d_c such that $\sum d_c = 1$
    Append d_c to $\mathcal{D}$
end for
Initialize $minKL \leftarrow +\infty$, $bestIndex \leftarrow -1$
for $i = 1$ to length of $\mathcal{D}$ **do**
    Compute $kl \leftarrow \mathrm{KL}(\mathcal{D}_i || \hat{\boldsymbol{y}})$
    if $kl < minKL$ **then**
        Update $minKL \leftarrow kl$, $bestIndex \leftarrow i$
    end if
end for
Set $\hat{\boldsymbol{l}} \leftarrow \mathcal{C}_{bestIndex}$

2.3.2 Intersubject Fine-tuning

We trained 40 models specific to the NinaPro DB2 database of healthy subjects and 11 specific to the Ninapro Db3 database of amputees by finetuning the previously obtained geometry-agnostic (pretrained) model. All models were trained in a using a leave-one-out cross-validation strategy: For Ninapro DB2, each finetuned model was thus trained on 39 subjects, excluding the subject that was to be used later as test data. Similarly, each model finetuned on Ninapro DB3 was trained on 10 subjects.

As each model was intended to operate under a single measurement geometry, there was no need for a fully generalized mapping from electrode geometry to positional embeddings. Instead, a learnable 10×1024 matrix was used as the value of $\boldsymbol{PE}$. This matrix was initialized using the output of the position-encoding network (described in section 2.2) when fed the manually

estimated 3D positions of the 10 electrodes, but was thereafter optimized together with the rest of the model. Naturally, the remaining parameters of the models were initialized as those learned during pretraining

During finetuning training, the classifier head (as opposed to the multi-label head) with softmax was used to generate model outputs. With one-hot encoded ground-truth targets $\boldsymbol{y}$, the loss to be minimized was the standard multiclass cross-entropy:

$$\mathcal{L}_{finetune} = \sum_{j=1}^{18} D_{KL}(y_i||\hat{y}_i) \tag{4}$$

where $\hat{y}_i$ represents the ith softmax output. The Adam algorithm with $\beta_1 = 0.9$, $\beta_2 = 0.999$ was once again used, this time with a smaller batch size of 512, to minimize $\boldsymbol{\mathcal{L}}_{finetune}$. The finetuning process lasted for 10000 iterations, using constant learning rate of $\eta_{base} = 0.0001$.

2.4 Performance Evaluation

Due to its paradigmatic role in myoelectric motor decoding, specifically for prosthesis control applications, we use the Linear Discriminant Analysis (LDA) algorithm as a intrasubject benchmark with which to compare both the zero-shot and finetuned intersubject models. For both Ninapro DB2 and DB3, the TD4 feature set (Mean Absolute Value, Waveform Length, Zero Crossings, and Slope Sign Changes) was extracted channel-wise from the same signal windows as obtained from the previous segmentation (200 ms window duration, 50 ms overlap). Once per subject, feature vectors originating from the first 3 movement repetitions were used to compute summary statistics (for the purpose of standardization to zero mean and unit variance) and train subject-specific models, while the last 3 repetitions were withheld for testing.

The pretrained model used Algorithm 2 (together with the label decompositions of the 17 movement classes) to make inferences during testing—as LDA implicitly assumes that only the (in this case 17) classes seen during training are possible, this is arguably a more fair comparison than using Algorithm 1 would be. Both the pretrained model and the finetuned models was evaluated on (the raw sEMG data of, not the features of) the same test data (comprising repetitions 4 to 6) as the LDA models.

For all three approaches (user-calibrated LDA, fully general ANN using zero-shot inference, and database-finetuned ANNs using end-to-end inference), we report database-wide classification accuracy (fraction of correct inferences to total number of inferences) as the measure of model performance.

3 Results and Discussion

On a desktop computer equipped with an NVIDIA 3090 GTX GPU, a single forward-backward pass of a batch through the model required a wall time of 13.1 seconds, corresponding to a total wall time of approximately 20 days for the pretraining and a total wall time of approximately 36 h for each of the $40 + 11 = 51$ finetuned models. A single forward pass of a single example with 10 sEMG channels lasted for a duration of 70.3 ms. In and of itself, this delay would be compatible with common numbers given as the acceptable limit of myoelectric control[38]. However, the computational capabilities of the training setup used in this study are not at all representative of those found in the embedded hardware of, for example. upper-limb prosthetics.

An important future research direction to make large sEMG decoding models of the kind introduced in this study viable would thus be one of *model distillation*, i.e. techniques for reducing the computational requirements of already trained models. Many previous studies have approached and proposed solutions to this specific problem (see e.g.[39, 40, 41]) , some of which could be implemented for sEMG motor decoding ANNs without significant modification

Class-wise test accuracies for all models of this study on the Ninapro DB2 database of healthy subjects are shown in the confusion matrices of Figure 2. The test set accuracy (across all subjects and movement classes) exhibited by LDA was 53.1%. While this number is significantly lower than what is usually reported in the literature, it is important to keep in mind that the rest class was excluded. As such, the test sets were all almost perfectly balanced, and the most easily detectable class absent. For the pretrained model using zero-shot inference and intersubject finetuned models making end-to-end inferences the accuracies were 24.7% and 50.9%, respectively. While the pretrained model underperforms the benchmark substantially, it produces results far better than those expected from random guessing. This indicates the feasibility of not only inter-user models, but fully general motor intent decoding models independent of specific geometries. Perhaps more importantly, intersubject models achieve approximately the same accuracy as LDA, despite never having observed data from the subject during training. In light of this finding, we suggest that the method outlined in this paper could represent an path towards fully calibration-free myoelectric interfaces.

The test set accuracies (across all subjects) obtained on the Ninapro DB3 database of amputees were 34.8% for LDA, 7.7% for the pretrained model using zero-shot inference, and 9.7% for the database-finetuned ANN making end-to-end inferences. Class-wise accuracies are shown in the confusion matrices of Figure 3. Thus, whereas the aforementioned results on healthy users are promising, the performance on amputees is far less impressive. A likely explanation for this is the major difference in the spatial configuration of muscles in the forearm between healthy and amputee subjects[42].The same coordinate on the forearm of an amputee and healthy subject might not even correspond to the same underlying muscle(s)[43]. As the pretrained model of this study is optimized to extract instrumentally useful signal representations from exclusively intact subjects, this difference makes data from amputees far out of distribution for the models. Future research could aim to remedy this shortcoming, for example by introducing alternative positional encoding schemes that do not explicitly rely on cylinder surface coordinates. A potential step in this direction could be to represent the positions of electrodes via their position relative to muscles instead of globally on the forearm. In this way, a form of transfer learning between the signal domain of healthy users and amputee users could be obtained.

Figures

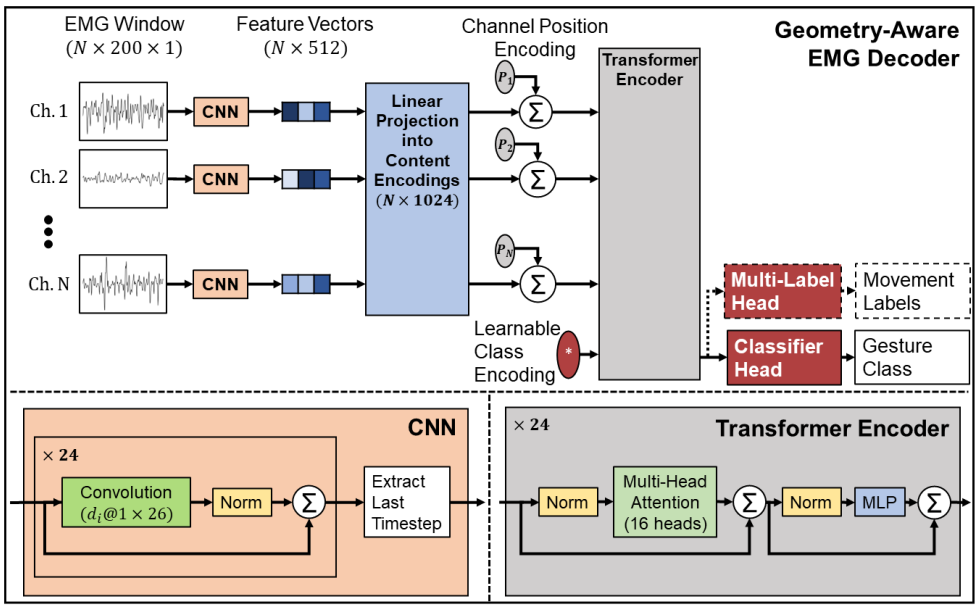
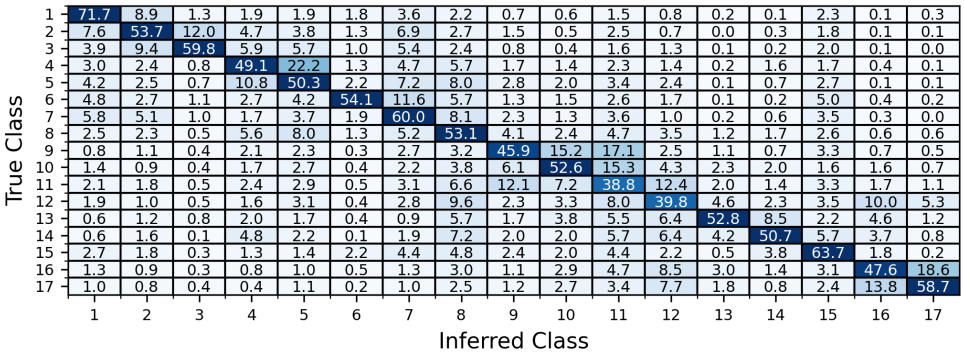
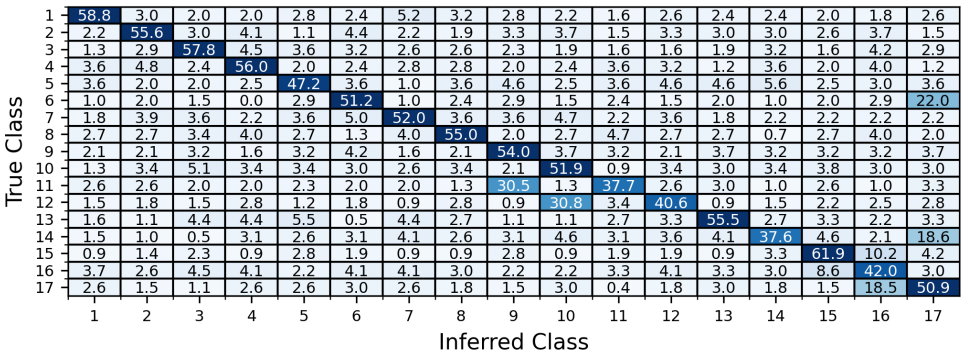


Figure 1: Overview of the architecture of the motion decoder neural network model. The $N \times 200$ input EMG window (where the number of channels N and their spatial distribution is allowed to vary between inputs) is first fed channel-wise through a temporal 1D CNN in order to obtain channel-specific feature vectors for each of the N channels. Feature vectors are then linearly embedded into fixed-length representations (content encodings), positionally encoded in accordance to the positions of their respective electrodes, and fed through a Transformer encoder model [20]. Like in a standard Vision Transformer [29], the transformer is fed an additional learnable token whose value following processing by the transformer module is fed into either a classifier head or a multi-label head, depending on the phase of the training.)

(a) User-calibrated LDA



(b) Finetuned model



(c) Zero-shot model

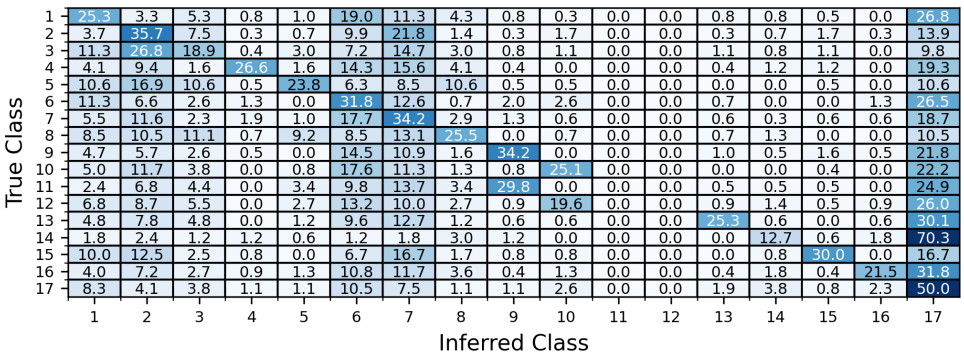
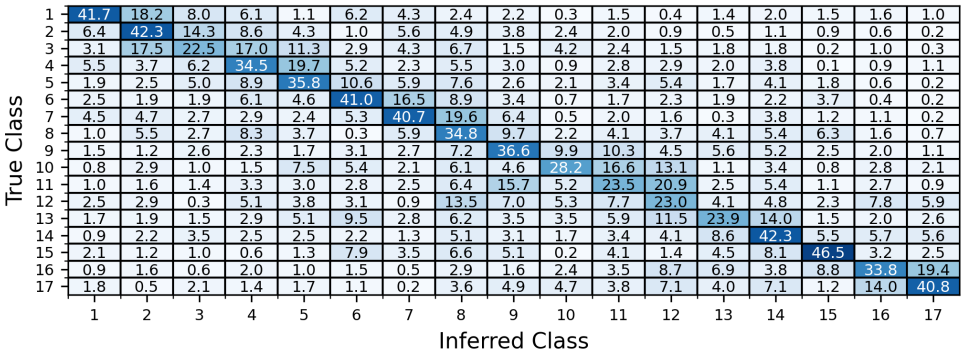
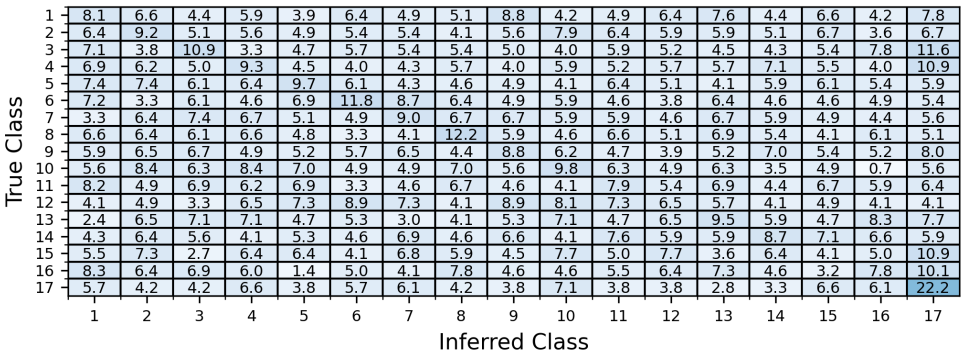


Figure 2: Confusion matrices of the two calibration-free approaches (zero-shot and finetuned) and the reference user-calibrated LDA method on the exercise B sub-database of the Ninapro DB2 database of healthy subjects. Values are aggregated across all 40 subjects.

(a) User-calibrated LDA



(b) Finetuned model



(c) Zero-shot model

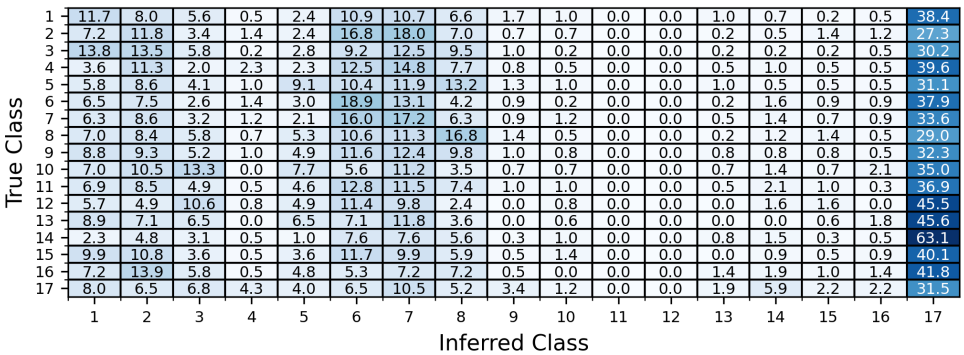


Figure 3: Confusion matrices of the two calibration-free approaches (zero-shot and finetuned) and the reference user-calibrated LDA method on the exercise B sub-database of the Ninapro DB3 database of amputees. Values are aggregated across all 11 subjects.

240

Tables

Table 1: Overview of the 30 EMG datasets used for training the geometry-aware, inter-user EMG motion decoding model.

Name	EMG channels	Unique motions	Participants	Processed size (GiB)
3DC [44]	10	11	22	1.5
Long-term 3DC [45]	10	11	20	1.8
Biopatrec 2013 [46]	4	11	20	0.12
Biopatrec 2014 [47]	8	27	17	0.52
Biopatrec 2015 [48]	4	10	8	0.15
CapgMyo DB-a [49]	128	11	18	21.5
CapgMyo DB-b [49]	128	11	18	10.8
CapgMyo DB-c [49]	128	13	10	2.5
CSL hdsemg [50]	168	27	5	5.0
Grabmyo [51]	28	17	43	32.6
Huang *et al.* 2016 [52]	16	12	5	0.62
Hyser [34]	256	34	20	66.0
Khusbaba *et al.* 2012a [53]	2	10	8	0.11
Khusbaba *et al.* 2012b [54]	8	15	8	0.94
Khusbaba *et al.* 2013 [55]	8	14	8	0.44
Khusbaba *et al.* 2014 [35]	7	8	11	1.4
Khusbaba *et al.* 2016 [56]	6	6	12	0.80
Malesevic *et al.* 2021 [57]	128	65	20	65.4
MECLab [58]	8	6	30	7.0
MyoUP [59]	8	22	8	0.54
Nearlab [60]	10	8	11	0.72
Ninapro DB4 [61]	12	53	10	2.7
Ninapro DB5 [61]	16	53	10	3.1
Ninapro DB6 [62]	14	53	9	10.2
Ninapro DB7 [63]	12	41	20	2.3
Ninapro DB8 [64]	16	19	10	5.9
Olsson *et al.* 2021 [65]	8	8	20	0.32
Ozdemir *et al.* 2022 [66]	4	10	40	0.72
PutEMG [67]	24	8	44	10.0
SEEDS [68]	126	13	25	125.2
Total	**4-256**	**108**	**510***	**380.6**

*Some participants are likely present in multiple databases.

Table 2: Basis labels used for generalized movement intent encoding.

Label index	Name
1	Little finger flexion
2	Little finger extension
3	Ring finger flexion
4	Ring finger extension
5	Middle finger flexion
6	Middle finger extension
7	Index finger flexion
8	Index finger extension
9	Thumb flexion
10	Thumb extension
11	Thumb abduction
12	Thumb adduction
13	Wrist flexion
14	Wrist extension
15	Wrist pronation
16	Wrist supination
17	Ulnar deviation
18	Radial deviation

Table 3: The movements classes of Ninapro DB2 and DB3, exercise B.

Class number	Movement description	Basis label decomposition
1	Thumb up	[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
2	Extension of index and middle finger, flexion of the others	[1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
3	Flexion of ring and little finger, extension of the others	[1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
4	Opposing base of little finger	[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0]
5	Extension of all fingers	[0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
6	Flexion of all fingers	[1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
7	Index extension, flexion of others	[1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
8	Adduction of extended fingers	[0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
9	Wrist supination (axis: middle finger)	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]
10	Wrist pronation (axis: middle finger)	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]
11	Wrist supination (axis: little finger)	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]
12	Wrist pronation (axis: little finger)	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]
13	Wrist flexion	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]
14	Wrist extension	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]
15	Radial deviation	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]
16	Ulnar deviation	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]
17	Wrist extension with closed hand	[1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0]

References

[1] R. Merletti and D. Farina, *Surface electromyography: physiology, engineering, and applications*. John Wiley & Sons, 2016.

[2] D. Farina and A. Holobar, "Human machine interfacing by decoding the surface electromyogram [life sciences]," *IEEE signal processing magazine*, vol. 32, no. 1, pp. 115–120, 2014.

[3] A. Chowdhury, R. Ramadas, and S. Karmakar, "Muscle computer interface: A review," *ICoRD'13: Global Product Development*, pp. 411–421, 2013.

[4] J. L. Contreras-Vidal, N. A. Bhagat, J. Brantley, J. G. Cruz-Garza, Y. He, Q. Manley, S. Nakagome, K. Nathan, S. H. Tan, F. Zhu, *et al.*, "Powered exoskeletons for bipedal locomotion after spinal cord injury," *Journal of neural engineering*, vol. 13, no. 3, p. 031001, 2016.

[5] E. Scheme and K. Englehart, "Electromyogram pattern recognition for control of powered upper-limb prostheses: state of the art and challenges for clinical use.," *Journal of Rehabilitation Research & Development*, vol. 48, no. 6, 2011.

[6] J. Ning, S. Dosen, K. Muller, and D. Farina, "Myoelectric control of artificial limbs—is there a need to change focus?," *IEEE Signal Process Mag*, vol. 29, no. 5, pp. 152–0, 2012.

[7] J. Paciga, P. Richard, and R. Scott, "Error rate in five-state myoelectric control systems," *Medical and Biological Engineering and Computing*, vol. 18, pp. 287–290, 1980.

[8] N. Nazmi, M. A. Abdul Rahman, S.-I. Yamamoto, S. A. Ahmad, H. Zamzuri, and S. A. Mazlan, "A review of classification techniques of emg signals during isotonic and isometric contractions," *Sensors*, vol. 16, no. 8, p. 1304, 2016.

[9] A. Ameri, M. A. Akhaee, E. Scheme, and K. Englehart, "Regression convolutional neural network for improved simultaneous emg control," *Journal of neural engineering*, vol. 16, no. 3, p. 036015, 2019.

[10] M. B. I. Reaz, M. S. Hussain, and F. Mohd-Yasin, "Techniques of emg signal analysis: detection, processing, classification and applications," *Biological procedures online*, vol. 8, pp. 11–35, 2006.

[11] S. Jiang, B. Lv, W. Guo, C. Zhang, H. Wang, X. Sheng, and P. B. Shull, "Feasibility of wrist-worn, real-time hand, and surface gesture recognition via semg and imu sensing," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3376–3385, 2017.

[12] T. Matsubara and J. Morimoto, "Bilinear modeling of emg signals to extract user-independent features for multiuser myoelectric interface," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 8, pp. 2205–2213, 2013.

[13] R. N. Khushaba, "Correlation analysis of electromyogram signals for multiuser myoelectric interfaces," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, no. 4, pp. 745–755, 2014.

[14] U. Cote-Allard, C. L. Fall, A. Drouin, A. Campeau-Lecours, C. Gosselin, K. Glette, F. Laviolette, and B. Gosselin, "Deep learning for electromyographic hand gesture signal classification using transfer learning," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 27, pp. 760–771, Apr. 2019.

[15] E. Campbell, A. Phinyomark, and E. Scheme, "Deep cross-user models reduce the training burden in myoelectric control," *Frontiers in Neuroscience*, vol. 15, p. 657958, 2021.

[16] D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. Van Der Maaten, "Exploring the limits of weakly supervised pretraining," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 181–196, 2018.

[17] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, *et al.*, "Improving language understanding by generative pre-training," 2018.

[18] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[19] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017.

[21] M. Atzori, A. Gijsberts, C. Castellini, B. Caputo, A.-G. M. Hager, S. Elsig, G. Giatsidis, F. Bassetto, and H. Müller, "Electromyography data for non-invasive naturally-controlled robotic hand prostheses," *Scientific Data*, vol. 1, pp. 2052–4463, 2014.

[22] R. E. Crochiere and L. R. Rabiner, *Multirate digital signal processing*, vol. 18. Prentice-hall Englewood Cliffs, NJ, 1983.

[23] W. Geng, Y. Du, W. Jin, W. Wei, Y. Hu, and J. Li, "Gesture recognition by instantaneous surface EMG images," *Scientific Reports*, vol. 6, Nov. 2016.

[24] A. E. Olsson, P. Sager, E. Andersson, A. Björkman, N. Malešević, and C. Antfolk, "Extraction of multi-labelled movement information from the raw hd-semg image with time-domain depth," *Scientific reports*, vol. 9, no. 1, pp. 1–10, 2019.

[25] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[26] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv preprint arXiv:1606.08415*, 2016.

[27] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[29] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: transformers for image recognition at scale," in *ICLR*, 2021.

[30] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[31] A. Joulin, L. van der Maaten, A. Jabri, and N. Vasilache, "Learning visual features from large weakly supervised data," in *Computer Vision – ECCV 2016*, pp. 67–84, Springer International Publishing, 2016.

[32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[33] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, JMLR Workshop and Conference Proceedings, 2010.

[34] X. Jiang, X. Liu, J. Fan, X. Ye, C. Dai, E. A. Clancy, M. Akay, and W. Chen, "Open access dataset, toolbox and benchmark processing results of high-density surface electromyogram recordings," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 29, pp. 1035–1046, 2021.

[35] R. N. Khushaba, M. Takruri, J. V. Miro, and S. Kodagoda, "Towards limb position invariant myoelectric pattern recognition using time-dependent spectral features," *Neural Networks*, vol. 55, pp. 42–58, 2014.

[36] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[37] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, June 2016.

[38] T. R. Farrell and R. F. Weir, "The optimal controller delay for myoelectric prostheses," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 15, pp. 111–118, Mar. 2007.

[39] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015.

[40] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *International conference on machine learning*, pp. 10347–10357, PMLR, 2021.

[41] Y. Wei, H. Hu, Z. Xie, Z. Zhang, Y. Cao, J. Bao, D. Chen, and B. Guo, "Contrastive learning rivals masked image modeling in fine-tuning via feature distillation," *arXiv preprint arXiv:2205.14141*, 2022.

[42] C. Castellini, A. E. Fiorilla, and G. Sandini, "Multi-subject/daily-life activity emg-based control of mechanical hands," *Journal of neuroengineering and rehabilitation*, vol. 6, no. 1, pp. 1–11, 2009.

[43] J. E. Sanders and S. Fatone, "Residual limb volume change: Systematic review of measurement and management," *The Journal of Rehabilitation Research and Development*, vol. 48, no. 8, p. 949, 2011.

[44] U. Côté-Allard, G. Gagnon-Turcotte, A. Phinyomark, K. Glette, E. Scheme, F. Laviolette, and B. Gosselin, "A transferable adaptive domain adversarial neural network for virtual reality augmented emg-based gesture recognition," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 29, pp. 546–555, 2021.

[45] U. Côté-Allard, G. Gagnon-Turcotte, A. Phinyomark, K. Glette, E. Scheme, F. Laviolette, and B. Gosselin, "Long-term 3dc dataset," 2019.

[46] M. Ortiz-Catalan, R. Brånemark, and B. Håkansson, "Biopatrec: A modular research platform for the control of artificial limbs based on pattern recognition algorithms," *Source Code for Biology and Medicine*, vol. 8, pp. 1751–0473, 2013.

[47] M. Ortiz-Catalan, B. Håkansson, and R. Brånemark, "Real-time and simultaneous control of artificial limbs based on pattern recognition algorithms," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, pp. 756–764, 2014.

[48] E. Mastinu, M. Ortiz-Catalan, and B. Håkansson, "Analog front-ends comparison in the way of a portable, low-power and low-cost emg controller based on pattern recognition," in *37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 2111–2114, 2015.

[49] Y. Du, W. Jin, W. Wei, Y. Hu, and W. Geng, "Surface EMG-based inter-session gesture recognition enhanced by deep domain adaptation," *Sensors*, vol. 17, p. 458, 2017.

[50] C. Amma, T. Krings, J. Böer, and T. Schultz, "Advancing muscle-computer interfaces with high-density electromyography," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 929–938, Association for Computing Machinery, 2015.

[51] N. Jiang, A. Pradhan, and H. Jiayuan, "Gesture recognition and biometrics electromyogram (grabmyo) (version 1.0.0)," *PhysioNet*, 2022. https://doi.org/10.13026/rtfc-np50.

[52] H. Huang, T. Li, C. Bruschini, C. Enz, V. M. Koch, J. Justiz, and C. Antfolk, "Emg pattern recognition using decomposition techniques for constructing multiclass classifiers," in *2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, pp. 1296–1301, 2016.

[53] R. N. Khushaba, S. Kodagoda, M. Takruri, and G. Dissanayake, "Toward improved control of prosthetic fingers using surface electromyogram (emg) signals," *Expert Systems with Applications*, vol. 39, no. 12, pp. 10731–10738, 2012.

[54] R. N. Khushaba and S. Kodagoda, "Electromyogram (emg) feature reduction using mutual components analysis for multifunction prosthetic fingers control," in *2012 12th International Conference on Control Automation Robotics & Vision (ICARCV)*, pp. 1534–1539, 2012.

[55] R. N. Khushaba, S. Kodagoda, D. Liu, and G. Dissanayake, "Muscle computer interfaces for driver distraction reduction," *Computer Methods and Programs in Biomedicine*, vol. 110, no. 2, pp. 137–149, 2013.

[56] R. N. Khushaba, A. Al-Timemy, S. Kodagoda, and K. Nazarpour, "Combined influence of forearm orientation and muscular contraction on emg pattern recognition," *Expert Systems with Applications*, vol. 61, pp. 154–161, 2016.

[57] N. Malesevic, A. Olsson, P. Sager, E. Andersson, C. Cipriani, M. Controzzi, A. Björkman, and C. Antfolk, "A database of high-density surface electromyogram signals comprising 65 isometric hand gestures," *Scientific Data*, vol. 8, 02 2021.

[58] A. Chan and G. Green, "Myoelectric control development toolbox," 01 2007.

[59] N. Tsagkas, P. Tsinganos, and A. Skodras, "Myoup," 2019.

[60] R. Soroushmojdehi, S. Javadzadeh, A. Pedrocchi, and M. Gandolla, "Transfer learning in hand movement intention detection based on surface electromyography signals," *Frontiers in Neuroscience*, vol. 16, 2022.

[61] S. Pizzolato, L. Tagliapietra, M. Cognolato, M. Reggiani, H. Müller, and M. Atzori, "Comparison of six electromyography acquisition setups on hand movement classification tasks," *PLoS ONE*, vol. 12, no. e0186132, pp. 2052–4463, 2017.

[62] F. Palermo, M. Cognolato, A. Gijsberts, B. Caputo, H. Müller, and M. Atzori, "Analysis of the repeatability of grasp recognition for hand robotic prosthesis control based on semg data," in *IEEE International Conference on Rehabilitation Robotics*, 2017.

[63] A. Krasoulis, I. Kyranou, M. S. Erden, K. Nazarpour, and S. Vijayakumar, "Improved prosthetic hand control with concurrent use of myoelectric and inertial measurements," *Journal of NeuroEngineering and Rehabilitation*, vol. 14, pp. 1743–0003, 2017.

[64] A. Krasoulis, S. Vijayakumar, and K. Nazarpour, "Effect of user practice on prosthetic finger control with an intuitive myoelectric decoder," *Frontiers in Neuroscience*, vol. 13, 2019.

[65] A. E. Olsson, N. Malesevic, A. Björkman, and C. Antfolk, "Learning regularized representations of categorically labelled surface emg enables simultaneous and proportional myoelectric control," *Journal of NeuroEngineering and Rehabilitation*, vol. 18, 2021.

[66] M. A. Ozdemir, D. H. Kisa, O. Guren, and A. Akan, "Dataset for multi-channel surface electromyography (semg) signals of hand gestures," *Data in Brief*, vol. 41, p. 107921, 2022.

[67] P. Kaczmarek, T. Mańkowski, and J. Tomczyński, "putemg-a surface electromyography hand gesture recognition dataset," *Sensors*, vol. 19, no. 16, 2019.

[68] A. Matran-Fernandez, I. J. Rodríguez Martínez, R. Poli, C. Cipriani, and L. Citi, "Seeds, simultaneous recordings of high-density emg and finger joint angles during multiple hand movements," *Scientific Data*, vol. 6, no. 186, 2019.

Paper VII

Exploiting the intertemporal structure of the upper-limb sEMG: comparisons between an LSTM network and cross-sectional myoelectric pattern recognition methods

A. Olsson, N. Malešević, A. Björkman, C. Antfolk

In *41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC) 2019*

Exploiting the Intertemporal Structure of the Upper-Limb sEMG: Comparisons between an LSTM Network and Cross-Sectional Myoelectric Pattern Recognition Methods

Alexander Olsson, Nebojša Malešević, Anders Björkman, and Christian Antfolk

Abstract— **The use of natural myoelectric interfaces promises great value for a variety of potential applications, clinical and otherwise, provided a computational mapping between measured neuromuscular activity and executed motion can be approximated to a satisfactory degree. However, prevalent methods intended for such decoding of movement intent from the surface electromyogram (sEMG) based on pattern recognition typically do not capitalize on the inherently time series-like nature of the acquired signals. In this paper, we present the results from a comparative study in which the performances of traditional cross-sectional pattern recognition methods were compared with that of a classifier built on the natural assumption of temporal ordering by utilizing a long short-term memory (LSTM) neural network. The resulting evaluation indicate that the LSTM approach outperforms traditional gesture recognition techniques which are based on cross-sectional inference. These findings held both when the LSTM classifier operated on conventional features and on raw sEMG and for both healthy subjects and transradial amputees.**

I. Introduction

The surface electromyogram (sEMG) [1] acquired from the human forearm contains a significant amount of information concerning neuromuscular activity associated with the evolving state of the hand and wrist during muscle usage. In essence, every channel of the sampled sEMG time series represents a superposition of motor unit action potentials (MUAP) originating from motor units located sufficiently close to its respective electrode pickup-site. Systems aimed at solving the nontrivial problem of untangling this seemingly stochastic, nonstationary and nonlinear mixture of bioelectrical factors in order to extract actionable commands are typically referred to as muscle-computer interfaces (MCU) and have been subject to the attention of the research community for some time now. Current and potential future use cases for direct interfacing of this kind include steering of exoskeletons [2], video gaming [3], and, likely most commonly, control of upper limb prosthetics.

This class of prostheses; controlled via voluntary neuromuscular activity in the residual limb, are typically referred to as *myoelectric* and have been a part of the clinical repertoire for a number of decades [4]. However, the more recent emergence of technologies based on *pattern recognition* has ignited the search for devices which are controlled *naturally*: that is to say, the attempted performance of a motion by the user corresponds to MCI output encoding the very same movement. Such devices have the potential to greatly alleviate the troubles faced by upper limb amputees, whose quality of life more often than not lessens after the loss of a limb due to the notable decrease of functionality accompanying the traumatic event [5].

Hitherto proposed systems for natural and real-time myoelectric control based on pattern recognition typically follow a similar structure, namely that of initial sEMG acquisition and signal preprocessing, followed by window-based feature extraction and culminating in a single multi-class inference per feature window [6]. Whereas very impressive classification accuracies in the excess of 95% have been reached with such schemes [7] (although with sets of performable movements somewhat limited in cardinality), widespread clinical adoption has not yet been realized. Although they offer significantly more functionality than the status quo, pattern recognition-based method still lacks the stability and reliability required for deployment outside of the laboratory. Much work is thus still required on this front.

Traditional classifiers which have oftentimes been utilized as the final step within the classificatory framework outlined above have been *cross-sectional* in nature. Here, cross-sectional is taken to mean that the classification of one feature window is not impacted by the contents of earlier parts of the acquired sEMG signal. Thus, while sensitive to the temporal structure *within* each feature extraction window (granted a sufficiently sophisticated feature set and/or classifier), methods of this kind by necessity disregard the intertemporal relationship *between* windows. Trivially, in time series-like signals with strong intertemporal structure, cross-sectional classifiers as such by definition forfeit a wealth of information with potentially high predictive power. We hypothesize that allowing information to flow forwards in time to implicitly adjust the Bayesian prior probabilities at the moment of each classification decision should have a favorable effect on observed system performance.

A recurrent neural network (RNN) [8] is a type of classifier which possess this desired property of allowing information from previous signal observations to impact later decisions. This class of methods has become ubiquitous in many applications of digital signal processing but has not until very recently been introduced into the context of myoelectric decoding, with highly promising results [9,10]. To further explore this avenue of inquiry, this paper presents a comparative study in which a traditional myoelectric pattern recognition pipeline incorporating cross-sectional

*Alexander Olsson, Nebojša Malešević and Christian Antfolk are with the Dept. of Biomedical Engineering, Faculty of Engineering, Lund University. P.O. Box 118, Lund, Sweden. {alexander.olsson, nebojsa.malesevic, christian.antfolk} @bme.lth.se. Anders Björkman is with the Department of Translational Medicine – Hand Surgery, Skåne University Hospital and Lund University, Jan Waldenströms gata 5, SE-205 03 Malmö, Sweden (anders.bjorkman@med.lu.se). Correspondance to Christian Antfolk. This work was supported by the Crafoord Foundation, the Promobilia Foundation and the European Commission under the DeTOP project (LEIT-ICT-24-2015, GA #687905).

978-1-5386-1311-5/19/$31.00 ©2019 IEEE

classifiers is evaluated alongside a pipeline with a to the problem novel long short-term memory (LSTM) [8] recurrent neural network classifier. Results are presented in the form of relevant performance measures computed sEMG data collected from both intact and amputee test subjects.

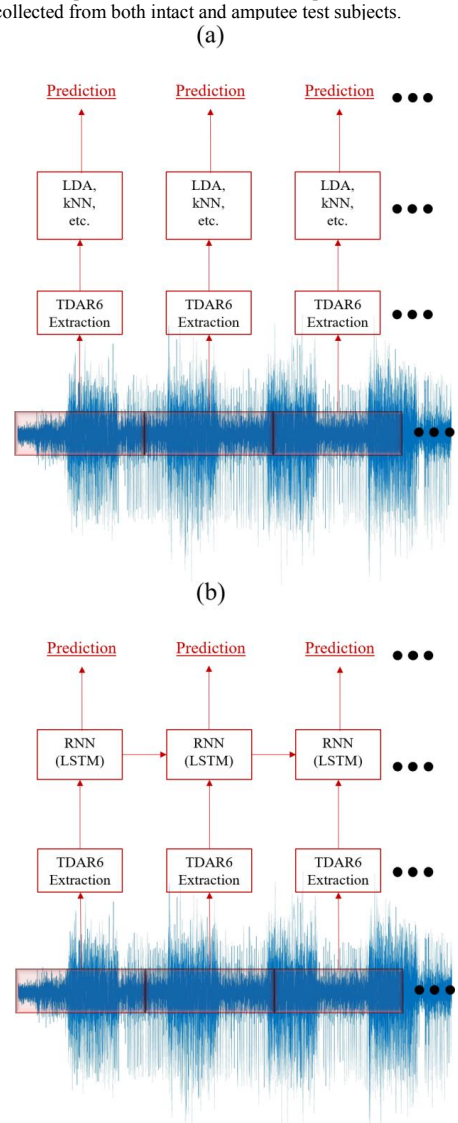


Figure 1. Visualization of the myoelectric pattern recognition procedure for (a) the cross-sectional classifiers and (b) the LSTM network time series classifier (when applied to extracted TDAR6 features). Note that the LSTM-based classifier was additionally applied directly to raw sEMG measurements (not pictured).

II. METHODS

A. Data acquisition

The sEMG recording, with corresponding gesture annotations, which were processed (offline) in this study were collected from the publicly available NinaPro dataset of hand gestures [11]. Three separate databases in total; DB2 [11], DB3 [11] and DB7 [12], were selected for the purposes of our analysis. DB2 contains sEMG recordings acquired from 40 intact and otherwise healthy test subjects. DB3 contains sEMG recordings acquired from 13 transradial amputees. DB7 contains sEMG recordings acquired from 22 test subject, of which 20 are healthy and 2 are amputees. As such, the data under consideration here represents a total of 60 intact subjects and 13 amputees with varying amount of residual limb remaining. We concentrated our work exclusively on exercises A and B; identical across the three databases, containing within them both rest and 40 mutually exclusive isometric and isotonic postures, wrist movements and grasps, all performed by every subject for a total of 41 motion classes. During acquisition, each such motion is performed 6 times for a duration of 5 s each, with 3 s of rest separating each repetition. sEMG signals are acquired from 12 surface electrodes; 8 of which are equidistantly enclosing the forearm at the height of the humeroradial joint, 2 of which are situated at the flexor digitorum superficialis and the flexor digitorum superficialis muscles, respectively, and the 2 remaining covering the biceps brachii muscle and its antagonist; the triceps brachii muscle. The actual sEMG measurements thus take the form of 12 separate digital time series sampled at a rate of 2 kHz and a synchronously acquired target signal of integers encoding the motion cooccurring with sEMG acquisition.

B. Preprocessing

In addition to the preprocessing steps already applied to the externally collected data as part of the acquisition protocol (e.g. Hampel filtering), we separately filter each sEMG channel with both a 2:th order 48-52 Hz digital Butterworth band stop filter (for removal of power line interference) and with a 6:th order 5-500 Hz digital Butterworth band pass filter (for the removal of noise).

Each time sample is labeled according to its repetition affiliation, i.e. assigned an integer value between 1 and 6. Moments of rest are grouped together with the repetition closest to them in time. The signal can subsequently be segmented according to repetition affiliation, with those samples belonging to repetitions 1-4 grouped into a training sequence and those belonging to repetitions 5-6 into a validation sequence. This procedure generates a training and a validation sequence, each with an internal order congruent with that in which samples were originally acquired through time. Thus, as different segments of each sequence are in reality extracted from different motions and even recording sessions, this method will sometimes result in long 'jumps' in time between adjacent samples. However, as all discontinuities of this kind occur during the moments of rest between motions, we assume they do to not significantly impact the observable dynamics of the sEMG measurements.

With temporally structured, raw sEMG training and validation data now available for every subject, features are extracted via a sliding window of length 100 ms (200

samples at the sampling rate of 2 kHz) and with increment step size of 50 ms (100 samples, i.e. 50% window overlap). The sliding window is independently used to scan the training and validation sequences, thus giving rise to two new temporally structured sequences of feature values per subject. The selected feature set is composed of the 4 time-domain features pioneered by Hudgins *et al.* [13], i.e. MAV (mean average value), WL (waveform length), ZC (zero crossings) and SSC (slope sign changes), concatenated with the 6 characteristic polynomial coefficients of a to the samples of the window fitted 5:th order AR (autoregressive) model [14]. This particular selection of features is ubiquitous in the sEMG pattern recognition literature [15] and is typically referred to as TDAR6. With 12 concurrent sEMG channels, this extraction method generated a total of $12\times(4+6)=120$ numeric values per sliding window location. Prior to classifier training, all features were rescaled by subtraction by the training sequence (per feature) mean and division by the training sequence (per feature) standard deviation. To assign a ground truth class label to each sliding window location, majority voting across the ground truth classes of the constituent window time samples are undertaken to consistently handle windows covering transient parts (onset of motion and rest) of the signal.

C. Classification

Five well-known statistical classification algorithms, selected to represent the cross-sectional status quo, are used for benchmarking of the proposed LSTM-based alternative: LDA [16] (linear discriminant analysis), SVM [17] (support vector machine), kNN [18] (k-nearest neighbors), RF [19] (random forest) and ERT [20] (extremely randomized trees). These methods have all with varying degrees of success been applied to the problem of myoelectric pattern recognition in previous studies [15], with the RF classifier oftentimes acknowledged as well-adapted to tasks with large motion class set cardinality [11]. As hyperparameter optimization lies outside the scope of this study, hyperparameters were (when applicable) selected ad-hoc in accordance with what is typical in tasks of pattern recognition: For SVM, the kernel function is set as RBF (radial basis function). For kNN, the number of neighbors is set as 10. For both RF and ERT, the number of trees is set as 100. For all classifiers where applicable, the multiclass decision function is selected as *one-versus-rest*. For implementation[1], we utilized the open source scikit-learn library [21] (interfaced with via Python 3.6).

During training, the five cross-sectional classifiers are, once per subject, fitted to the training sequence of feature values (originating from motion repetitions 1, 2, 3 and 4) extracted as described in the previous sections. The training data is balanced by randomly discarding window locations labeled as belonging to more abundant classes until the number of available examples are equal for all motion classes. As the rest class is massively overrepresented in the collected data, this step is necessary to avoid classifier bias and later give meaningful performance metric values.

The titular recurrent neural model of this paper contains a single LSTM layer with 64 output nodes and a tanh output gate activation, followed by batch normalization [22] and 25% dropout [23], ending with a 41-way (the number of motion classes) fully connected layer and a SoftMax activation layer for classification. For training, we use the Adam algorithm [24] with learning rate 0.0005, $\beta_1=0.9$ and $\beta_2=0.999$ together with L_2 weight regularization with a decay rate of $\lambda=10^{-6}$. The truncated backpropagation through time (T-BPTT) scheme for optimization employed here makes use of a per training sample memory of 1000 time steps. The training procedure proceeded for a total of 50 epochs, with a minibatch size of 32. This resulted in an aggregate model training time of approximately 2h per test subject on a desktop equipped with a NVIDIA GeForce GTX 1070 GPU, with models instantiated[1] in Python 3.6 via TensorFlow [25].

Two separate LSTM networks of this kind are trained (subject-wise): one on the training sequence of feature windows, and one on the raw sEMG training sequence. For the latter case, the raw sEMG signals are downsampled by a factor of 2 due to memory restrictions and rescaled in the same fashion as the feature sequences. Just as for the cross-sectional classifiers, data is balanced w.r.t. class labels by randomly ignoring time points corresponding to overrepresented classes in proportion to relative class abundance. This ensures that all classes are used to an equal degree in the updating of network weights.

Following training, all classifiers were evaluated on the validation sequences (acquired from motion repetitions 5 and 6 for each subject). The validation sequences are, in contrast to the training sequences, not balanced, as interference of this kind would not preserve the temporal ordering required by the LSTM networks for proper functioning. The internal state of the LSTM cells is not reset following inference, and predictions are generated at each time point based on both the contemporary sample and the internal representation of the validation sequence in its entirety so far. As the LSTM network trained on raw sEMG signals thus produces one prediction per time sample, simple majority voting with a sliding window with the same properties as the feature extraction window (i.e. 200 ms in length, 50% overlap) is undertaken on its output in order to yield results identical in classification rate to those of the other methods.

D. Performance metrics

To evaluate and subsequently compare the manifest performance of the fitted classifiers, a set of metrics previously introduced in the EMG pattern recognition literature is computed for all subjects and motion repetitions (listed below). Each metric gives rise to a single descriptive scalar value per test subject.

- *Accuracy (Acc):* The proportion of sliding window locations (globally) which predicted class corresponds to their ground truth class. The optimal (highest possible) value for Acc is 100%.
- *Motion Selection Time (MS)* [26]*:* The average (across motion repetitions) number of window increments between motion onset and the first correctly classified window. Motion onset for a repetition is defined as the first sample whose corresponding ground truth class signal assumes a value different from rest. Repetitions where selection is never reached (i.e. the classifier never correctly infers the correct motion) are ignored for the purposes of computing MS. The optimal (lowest possible) value for MS is 0.

- *Motion Completion Time (MC)* [26]*:* The average (across motion repetitions) number of window increments between motion onset and motion completion. Motion completion for a repetition is here defined as the sliding window location where a total of 20 correct inferences have been made by the classifier (cumulatively). Repetitions where completion is never reached (i.e. less than 20 windows in total are classified correctly) are ignored for the purposes of computing this metric. The optimal (lowest possible) value for MC is 20.
- *Motion Completion Rate (MCR)* [26]*:* The proportion of repetitions (globally) for which motion completion is reached. The optimal (highest possible) value for MCR is 100%.
- *Dynamic Efficacy (DE)* [26]*:* The ratio of correct predictions to total predictions (cumulative over the repetition) at the window at which motion completion is achieved, averaged across all repetitions. Repetitions where completion is never reached are ignored for the purposes of computing this metric. The optimal (highest possible) value for DE is 100%.

In addition, we compute two novel measures of performance as defined below:

- *Effective Accuracy (EA):* The product of MCR and DE. The optimal (highest possible) value for EA is 100%.
- *Motion Set Utility (MSU):* The proportion of motions classes for which *all* repetitions were completed successfully. The computation of this metric allows for evaluation of both the classifier in question and the selected set of movements, and furthermore identifies whether successful motion completions are concentrated to a limited set of motions or not. The optimal (highest possible) value for MSU is 100%.

III. Results

The performances of the cross-sectional classifiers (operating on the TDAR6 feature set) are presented in table 1. The performance of the LSTM time series classifier (operating on both the TDAR6 feature set and raw sEMG measurements) is presented in table 2. With healthy subjects, the highest accuracy was achieved with quite a large margin by the LSTM network model operating on the TDAR6 feature set (65.7±7.0%), followed closely by the very same classifier operating on raw sEMG voltages (64.0±12.8%). The advantages of these classifiers prevailed for amputee subjects, where their respective accuracies decreased to 48.2±13.0 and 37.7±11.9%. In the EMG-specific metrics, the RF and ERT classifiers are often similar to and sometimes even surpasses the performance of the LSTM network. As can be deduced from the ranges presented in the tables, intersubject variation is quite significant for many classifier-metric combinations.

	LDA	SVM	kNN	RF	ERT
Acc	Healthy: 49.7±12.8 Amputees: 28.0±11.3	Healthy: 52.8±11.9 Amputees: 29.7±11.6	Healthy: 41.7±13.7 Amputees: 22.9±9.9	Healthy: 53.0±13.9 Amputees: 29.6±13.2	Healthy: 52.6±14.3 Amputees: 29.4±13.1
MS	Healthy: 8.3±2.3 Amputees: 11.5±5.3	Healthy: 6.4±2.4 Amputees: 10.6±4.9	Healthy: 6.8±2.1 Amputees: 11.9±5.3	Healthy: 5.6±2.2 Amputees: 10.2±4.7	Healthy: 5.7±1.9 Amputees: 10.3±5.9
MC	Healthy: 40.2±6.0 Amputees: 51.6±8.2	Healthy: 38.3±5.5 Amputees: 52.3±10.1	Healthy: 42.9±6.3 Amputees: 57.3±7.9	Healthy: 35.9±5.2 Amputees: 50.0±10.0	Healthy: 36.8±5.4 Amputees: 51.4±10.3
MCR	Healthy: 93.3±3.4 Amputees: 65.9±18.2	Healthy: 94.9±3.3 Amputees: 69.0±17.1	Healthy: 88.8±6.0 Amputees: 54.5±18.0	Healthy: 95.7±3.3 Amputees: 71.7±17.9	Healthy: 95.4±3.6 Amputees: 69.3±18.8
DE	Healthy: 56.8±6.6 Amputees: 46.9±6.6	Healthy: 59.2±6.6 Amputees: 46.7±7.5	Healthy: 53.0±6.3 Amputees: 41.7±6.3	Healthy: 63.3±7.1 Amputees: 49.4±8.7	Healthy: 61.8±7.1 Amputees: 48.2±8.7
EA	Healthy: 53.0±6.6 Amputees: 31.5±10.5	Healthy: 56.2±6.9 Amputees: 33.0±9.9	Healthy: 47.1±6.6 Amputees: 23.3±8.9	Healthy: 60.6±7.1 Amputees: 36.5±11.6	Healthy: 59.0±7.3 Amputees: 34.5±11.4
MSU	Healthy: 89.3±5.2 Amputees: 54.3±18.0	Healthy: 91.4±5.0 Amputees: 56.7±15.5	Healthy: 83.0±8.4 Amputees: 42.6±16.0	Healthy: 92.7±5.2 Amputees: 60.6±16.3	Healthy: 92.4±5.4 Amputees: 57.9±17.8

Table 1. The performance metrics for each of the cross-sectional classifiers. The reported values were computed via averaging over all subjects across all databases. The reported ranges (following ±) represent the standard deviations (computed across all subjects).

	LSTM (on TDAR6 features)	LSTM (on raw sEMG)
Acc	Healthy: 65.1±9.3 Amputees: 43.8±12.1	Healthy: 64.0±12.8 Amputees: 37.7±11.9
MS	Healthy: 12.7±3.5 Amputees: 18.0±7.1	Healthy: 13.2±5.0 Amputees: 17.3±7.6
MC	Healthy: 34.3±4.0 Amputees: 39.6±7.7	Healthy: 36.6±5.3 Amputees: 40.8±8.5
MCR	Healthy: 81.9±6.1 Amputees: 63.5±14.1	Healthy: 91.3±5.4 Amputees: 64.9±20.7
DE	Healthy: 70.5±5.6 Amputees: 67.1±8.4	Healthy: 66.2±6.3 Amputees: 64.4±12.3
EA	Healthy: 57.9±7.1 Amputees: 43.5±12.4	Healthy: 60.6±7.8 Amputees: 39.7±12.7
MSU	Healthy: 72.2±8.4 Amputees: 49.6±13.1	Healthy: 85.9±8.0 Amputees: 52.3±16.8

Table 2. The performance metrics of the LSTM-based classifier. The reported values were computed via averaging over all subjects across all databases. The reported ranges (following ±) represent the standard deviations (computed across all subjects).

IV. Discussion and Conclusions

In this comparative study, five cross-sectional classification algorithms (LDA, SVM, kNN, RF, ERT) and one time-series based classification algorithm (an LSTM network) were evaluated on data from a total of 73 subject, collected from the publicly available NinaPro database. The time series approach resulted in significantly higher accuracy, without significant cost to the other selected performance metrics. However, compared to other studies utilizing similar methods, e.g. [11], and applying them to the same data, the performance presented here was notably worse. The reason for this is conjectured to lie mainly in our deliberate balancing of the training data, resulting in classifiers unable to capitalize on the abundance of rest samples in the validation data. Somewhat unexpectedly, the use of our LSTM network achieved slightly higher accuracy when applied to TDAR6 features than when applied to raw sEMG data. It is indeed possible that the combination of traditional feature extraction-based methods with more modern 'deep' approaches can yield better results when combined effectively than either applied separately. Regardless, our results indicate that approaches based on RNN topologies are a promising research topic to investigate further in the search for stable and accurate myoelectric interfaces.

References

[1] M. B. Reaz, M. Hussain, and F. Mohd-Yasin, "Techniques of EMG signal analysis: detection, processing, classification and applications", *Biol. Proced. Online*, vol. 8, pp 11-35, 2006.

[2] R. Jimenez-Fabian and O. Verlinden, "Review of control algorithms for robotic ankle systems in lower-limb orthoses, prostheses, and exoskeletons", *Med. Eng. Phys.,* vol 34, pp. 397-408, 2012.

[3] D. Kim, N. Lu, R. Ma, Y. Kim, R. Kim, S. Wang, J. Wu, S. Min Won, H. Tao, A. Islam, K. Yu, T. Kim, R. Chowdhury, M. Ying, L. Xu, M. Li, H. Chung, H. Keum, M. McCormick, and J. A. Rogers, "Epidermal electronics", *Science* vol 333, pp 838–843, 2011.

[4] S. Hubbard, "Myoprosthetic management of the upper limb amputee", in *Rehabilitation of the hand: Surgery and therapy, 4th ed.*, pp. 1241-1252, CRC Press, 1995.

[5] K. Demet, N. Martinet, F. Guillemin, J. Paysant, and J.M. André, "Health related quality of life and related factors in 539 persons with amputation of upper and lower limb", *Disabil. Rehabil.*, vol 25, 2003.

[6] A. D. Roche, H. Rehbaum, D. Farina, and O. C. Aszmann, "Prosthetic Myoelectric Control Strategies: A Clinical Perspective", *Curr. Surg. Rep.*, vol. 2, pp. 2-44, 2014.

[7] A. Phinyomark, P. Phukpattaranont, and C. Limsakul, "Feature reduction and selection for emg signal classification", *Exp. Sys. Appl.*, vol 39, pp. 7420–7431, 2012.

[8] S. Hochreiter and J. Schmidhuber, "Long short-term memory", *Neural Comput.,* vol. 9, pp. 1735–1780, 1997.

[9] Y. Hu, Y. Wong, W. Wei, Y. Du, M. Kankanhalli, and W. Geng, "A novel attention-based hybrid CNN-RNN architecture for sEMG-based gesture recognition", *PLoS ONE*, vol. 13, 2018.

[10] A. Samadani, "Gated Recurrent Neural Networks for EMG-Based Hand Gesture Classification: A Comparative Study" in *Conf. Proc. IEEE Eng. Med. Biol. Soc.*, 2018.

[11] M. Atzori, A. Gijsberts, C. Castellini, B. Caputo, A. G. M. Hager, S. Elsig, G. Giatsidis, F. Bassetto, and H. Müller, "Electromyography data for non-invasive naturally-controlled robotic hand prostheses", *Sci. data*, vol. 1, article number 140053, 2014.

[12] A. Krasoulis, I. Kyranou, M. S. Erden, K. Nazarpour, and S. Vijayakumar, "Improved prosthetic hand control with concurrent use of myoelectric and inertial measurements", *J. Neuroeng. Rehabil.,* vol 14, 2017.

[13] B. Hudgins, P. Parker, and R. N. Scott, "A new strategy for multifunction myoelectric control", *IEEE Trans. Biomed. Eng.*, vol. 40, pp. 82-94. 1993.

[14] A. Phinyomark, R. N. Khushaba, E. Ibáñez-Marcelo, A. Patania, E. J. Scheme, and G. Petri. "Navigating features: a topologically informed chart of electromyographic features space", *J. Royal Soc. Interface*, vol. 14, article number 20170734. 2017.

[15] M. Hakonen, H. Piitulainen, and A. Visala, "Current state of digital signal processing in myoelectric interfaces and related applications", *Biomed. Signal Proces.*, vol. 18, pp 334-359, 2015.

[16] T. Hastie, R. Tibshirani, and J. Friedman. "The elements of statistical learning", Springer New York, 2009.

[17] C. Cortes and V. Vapnik, "Support-vector networks", *Mach. Learn.*, vol 20, pp. 273-297, 1995.

[18] R. O. Duda, P. E. Hart, and D. G. Stork, "Pattern Classification", John Wiley & Sons, 2000.

[19] L. Breiman, "Random forests", *Mach. Learn.*, vol. 45, pp. 5-32, 2001.

[20] P. Geurts, D. Ernst, and L. Wehenkel, "Extemly randomized trees", *Mach. Learn.*, vol. 63, pp. 3-42, 2006.

[21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Scikit-learn: machine learning in Python", *J. Mach. Learn. Res.*, vol 12, pp. 2825-2830, 2011.

[22] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift" in *Proceedings of the ICML,* pp. 448-456, 2015.

[23] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting", *J Mach Learn Res,* vol. 15, pp. 1929-1958, 2014.

[24] D. Kingma and J. Ba, "Adam: A method for stochastic optimization", Preprint available: http://arxiv.org/abs/1412.6980, 2014.

[25] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: A system for large-scale machine learning" in *Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation*, 2016.

[26] Y. Geng, S. Oluwarotimi, Y. Wei and G. Li, "Improving the Robustness of Real-Time Myoelectric Pattern Recognition against Arm Position Changes in Transradial Amputees", *BioMed. Res. Int.*, Article ID 5090454, 2017.

[1]Source code is available from the authors upon reasonable request.

Paper VIII

Can deep synthesis of EMG overcome the geometric growth of training data required to recognize multiarticulate motions?

A. E. Olsson, N. Malešević, A. Björkman, C. Antfolk

In *43rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC) 2021*

2021 43rd Annual International Conference of the
IEEE Engineering in Medicine & Biology Society (EMBC)
Oct 31 - Nov 4, 2021. Virtual Conference

Can Deep Synthesis of EMG Overcome the Geometric Growth of Training Data Required to Recognize Multiarticulate Motions?*

Alexander E. Olsson[1], Nebojša Malešević[1], Anders Björkman[2], and Christian Antfolk[1]

***Abstract*— By being predicated on supervised machine learning, pattern recognition approaches to myoelectric prosthesis control require electromyography (EMG) training data collected concurrently with every detectable motion. Within this framework, calibration protocols for simultaneous control of multifunctional prosthetic hands rapidly become prohibitively long—the number of unique motions grows geometrically with the number of controllable degrees of freedom (DoFs). This paper proposes a technique intended to circumvent this combinatorial explosion. Using EMG windows from 1-DoF motions as input and EMG windows from 2-DoF motions as targets, we train generative deep learning models to *synthesize* EMG windows appertaining to multi-DoF motions. Once trained, such models can be used to complete datasets consisting of only 1-DoF motions, enabling simple calibration protocols with durations that scale linearly with the number of DoFs. We evaluated synthetic EMG produced in this way via a classification task using a database of forearm surface EMG collected during 1-DoF and 2-DoF motions. Multi-output classifiers were trained on either (I) real data from 1-DoF and 2-DoF motions, (II) real data from only 1-DoF motions, or (III) real data from 1-DoF motions appended with synthetic EMG from 2-DoF motions. When tested on data containing all possible motions, classifiers trained on synthetic-appended data (III) significantly outperformed classifiers trained on 1-DoF real data (II), although significantly underperformed classifiers trained on both 1- and 2-DoF real data (I) ($p < 0.05$). These findings suggest that it is feasible to model EMG concurrent with multiarticulate motions as nonlinear combinations of EMG from constituent 1-DoF motions, and that such modelling can be harnessed to synthesize realistic training data.**

I. INTRODUCTION

In order to restore upper limb functionality, a myoelectric prosthetic hand should ideally require little effort to control and at the same time allow the user to perform a wide range of grasps and motions. The first desideratum—naturalness of control—is in theory straightforwardly achievable by pursuing myoelectric control based on *pattern recognition* [1]. Within this framework, the prosthesis control problem is formulated as one of *statistical prediction*: by utilizing data comprised of electromyography (EMG) signals and concurrent motion labels, machine learning algorithms can be trained to *classify* EMG time segments as belonging to one out of multiple predefined motion classes. Decisions from algorithms trained in this way can be interpreted as motion commands to be sent to a motorized prosthesis.

By construction, pattern recognition myoelectric control requires EMG training data from every motion that is to be performable by the prosthesis user. This presents an obvious obstacle to achieving diversity of possible motions. The problem is exacerbated by the fact that *simultaneous control*—here meaning the ability to independently steer every kinematic degree of freedom (DoF) at the same time—is arguably one of the ultimate goals of the prosthesis control field [2]. To realize the scope of the problem of achieving simultaneous control with pattern recognition methods, one can consider the case of a prosthetic hand with D DoFs, each of which can assume S states. The number of unique motions M such a system can perform is $M = S^D$, i.e. the number of unique motions grows geometrically with the number of controllable DoFs. Thus, as the mechanical sophistication of prostheses increases, the number of motions that need to be recorded from amputees is subject to a combinatorial explosion. Corollarily, exhaustively recording EMG signals from every possible motion entails long and complicated acquisition protocols even for relatively small values of D and S.

In light of the difficulties in procuring sufficient data for training simultaneous control methods within the pattern recognition paradigm, we propose a method intended to circumvent the need for exhaustive, user-specific datasets. In brief, we introduce a novel artificial neural network (ANN) architecture and use it to explicitly model user-independent relationships between EMG from motions incorporating multiple DoFs and EMG from constituent 1-DoF motions. Once trained on pairings of EMG windows from a multi-subject dataset comprised of 1-DoF motions and all of their possible combination motions, such models can be used to *synthesize* user-specific EMG windows associated with multiarticulate motions from real examples of 1-DoF EMG windows. Using this framework, new prosthesis users would only need to perform all relevant 1-DoF motions (i.e. a total of $D \cdot S$ unique motions) for the purpose of calibration, after which standard pattern recognition control interfaces can be trained on a dataset including synthetic multi-DoF EMG. In this study, the quality of synthetic EMG produced with this method was evaluated via a classification task: performance metrics obtained from multi-output classifiers trained on real data was compared to metrics obtained from classifiers trained on real data augmented with synthetic EMG.

*Research supported by the Promobilia Foundation, the Crafoord Foundation, the European Commission under the DeTOP project (LEIT-ICT-24-2015, GA #687905), and the Swedish Research Council (DNR 2019-05601).

[1]A. E. Olsson, N. Malešević, and C. Antfolk are with the Dept. of Biomedical Engineering, Faculty of Engineering, Lund University. P.O. Box 118, Lund, Sweden. {alexander.olsson, nebojsa.malesevic, christian.antfolk}@bme.lth.se

[2]A. Björkman is with the Department of Hand Surgery, Institute of Clinical Sciences, Sahlgrenska Academy, University of Gothenburg, Sahlgrenska University Hospital, Gothenburg, Sweden. anders.bjorkman@gu.se

Previous work has been successful in synthesizing EMG signals for the purpose of training data augmentation, either via explicit simulations [3] or via deep generative learning [4]. Although studies concerning EMG motion decoding has often conceptualized motions as composed of 1-DoF 'basis' motions for the purpose of multi-output classification (e.g. [2], [5]), we are not aware of any existing attempts to leverage this combinatorial view in order to synthesize EMG.

II. METHODS

A. Data

The database of EMG recordings with synchronized motion annotations used here was originally collected for the purpose of a previous study [6]. In brief, myoelectric signals were recorded from 20 healthy subjects using a Myo armband (Thalmic labs, Canada) consisting of 8 circularly arranged bipolar surface electrodes. The armband was placed enclosing the dominant forearm of the subject at a level approximately 1/3 of the distance from the elbow to the wrist. EMG signals were sampled at a rate of 200 Hz. The collection protocol entailed the use of two DoFs: (I) flexion/extension of the wrist and (II) flexion/extension of all digits simultaneously. Motions were encoded using a ternary scheme, wherein each DoF could assume 1 out of 3 values at each time point: -1 (flex), 0 (neutral/stall), or 1 (extend). This scheme allows for $S^D = 3^2 = 9$ possible motions (listed in table I), all of which were recorded. Each motion was repeated $R = 3$ times, with repetitions lasting for 5 seconds and separated by 3 s of rest.

An inter-subject leave-one-out cross-validation design was used to partition the data for the purpose of training and evaluating the generative framework. At each iteration, data from 19 subjects were used to train a synthesizer model via the procedures outlined in the sections following. Data from the remaining, held-out subject were lastly used as the basis for intra-subject classifier training and testing for the purpose of evaluating the impact adding synthetic training data has on pattern recognition performance.

TABLE I: The 9 recorded motions comprising the database.

Motion Description	Ternary Encoding
Rest	[0, 0]
Flexion of the wrist	[-1, 0]
Extension of the wrist	[1, 0]
Flexion of the digits	[0, -1]
Extension of the digits	[0, 1]
Flexion of the wrist & Flexion of the digits	[-1, -1]
Flexion of the wrist & Extension of the digits	[-1, 1]
Extension of the wrist & Flexion of the digits	[1, -1]
Extension of the wrist & Extension of the digits	[1, 1]

B. Preparation of Synthesizer Training Data

At each cross-validation iteration, raw EMG time series $X_c[n]$ from the 19 subjects constituting the training data were clipped at the 1:st and 99:th percentiles and normalized to the range $[-1, 1]$ separately for each channel $c = 1, ..., 8$. Conditioned signals $S_c[n]$ obtained in this way were, separately for each subject, segmented into a set of EMG time windows $\{\mathbf{E}^r_{i,j}\}$, where $\mathbf{E}^r_{i,j} \in \mathbb{R}^{600\times 8}$ represents the middle 3 s (600 samples) of the r:th repetition of the motion with ternary encoding $i\in\{-1, 0, 1\}$, $j\in\{-1, 0, 1\}$. With 8 unique nonrest motions (from table I) repeated 3 times, $3\cdot 8 = 24$ such time windows were obtained per subject. To construct training data for the ANN model, EMG window instances originating from 2-DoF motions—designated as the *target* value—were paired with two EMG window instances originating from its two constituent, 1-DoF movements—designated as *input* values. By including every possible pairing of repetitions, this approach resulted in a training set consisting of a total of $M \cdot R^3 = 4 \cdot 3^3 = 108$ input-output pairs per subject, where $M = S^D - D - 1 = 4$ is the number of unique 2-DoF motions and $R = 3$ is the total number of available repetitions for each motion.

C. Synthesizer Model

All deep learning models used in this study were implemented and instantiated using the TensorFlow 1.12 library executed with Python 3.6. Inspired by the *variational autoencoding* [7] approach to distribution modeling, the synthesizer model architecture introduced and applied here (illustrated graphically in fig. 1) performs a mapping from two EMG windows recorded during two different single-DoF motions—DoF 1 and DoF 2, respectively—to an EMG window recorded concurrently with the motion consisting of DoF 1 and DoF 2 active simultaneously. Specifically, the architecture consists of 3 modules: two encoder networks, a mixer network, and a decoder network.

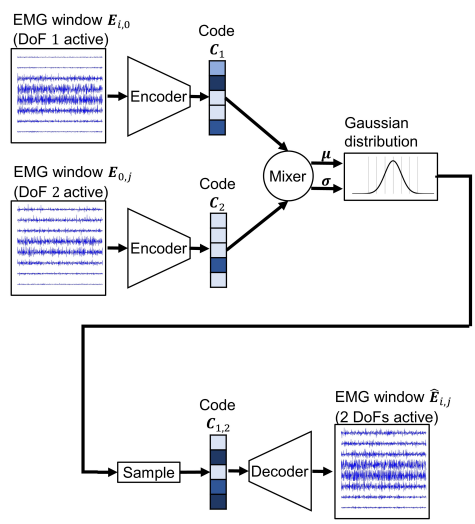


Fig. 1: Schematic overview of the synthesizer model.

- **Encoder**. The encoder networks transform the input 1-DoF EMG windows $\mathbf{E}_{i,0} \in \mathbb{R}^{600\times8}$ and $\mathbf{E}_{0,j} \in \mathbb{R}^{600\times8}$ into code vectors $\mathbf{C}_1 \in \mathbb{R}^{1024}$ and $\mathbf{C}_2 \in \mathbb{R}^{1024}$, respectively. They consist of 6 2D convolution layers with kernel sizes $[9\times1]$, $[1\times3]$, $[15\times1]$, $[1\times3]$, $[15\times1]$, and $[1\times3]$; output depths 64, 64, 256, 256, 1024, and 1024; and strides $[6\times1]$, $[1\times2]$, $[10\times1]$, $[1\times2]$, $[10\times1]$, and $[1\times2]$. Each layer is followed by leaky ReLU activation, layer normalization, and 50% dropout. The encoder networks share weights during training.
- **Mixer**. The mixer network transforms the encoder output codes $\mathbf{C}_1$ and $\mathbf{C}_2$ into a mean vector $\boldsymbol{\mu} \in \mathbb{R}^{1024}$ and standard deviation vector $\boldsymbol{\sigma} \in \mathbb{R}^{1024}$. This is achieved using a single fully connected layer of output size 2048 with linear activation function, whose output is split in two. The obtained vectors are used to define a 1024-dimensional distribution $N(\boldsymbol{\mu}, \text{diag}(\boldsymbol{\sigma}))$, from which a sample code $\mathbf{C}_{1,2}$ is drawn and presented as output.
- **Decoder**. The decoder network transforms an input mixture code $\mathbf{C}_{1,2} \in \mathbb{R}^{1024}$ into an 2-DoF synthetic EMG time window $\hat{\mathbf{E}}_{i,j} \in \mathbb{R}^{600\times8}$. Mirroring the architectures of the encoders, the decoder consists of 6 2D transposed convolution layers with kernel sizes $[1\times3]$, $[15\times1]$, $[1\times3]$, $[15\times1]$, $[1\times3]$, and $[9\times1]$; output depths 1024, 256, 256, 64, 64, and 1; and strides $[1\times2]$, $[10\times1]$, $[1\times2]$, $[10\times1]$, $[1\times2]$, and $[6\times1]$. The first 5 transposed convolution layer are followed by leaky ReLU activation, layer normalization, and 50% dropout; the final layer is followed by a linear activation.

Models were trained end-to-end to minimize the loss $\mathcal{L}$:

$$\mathcal{L} = \mathcal{L}_r + \mathcal{L}_s + \mathcal{L}_d \tag{1}$$

The *reconstruction* loss $\mathcal{L}_r$ quantifies the discrepancy between the synthetic 2-DoF EMG window produced by the network and the target 2-DoF EMG window provided during training. It is obtained by computing the squared Euclidean distance between the absolute frequency spectrum of the target $\mathbf{E_{i,j}}$ and the absolute frequency spectrum of the decoder output $\hat{\mathbf{E}}_{i,j}$:

$$\mathcal{L}_r = \|FFT(\mathbf{E}_{i,j})| - |FFT(\hat{\mathbf{E}}_{i,j})\|_2^2 \tag{2}$$

The *spread* loss $\mathcal{L}_s$ incentivizes the encoder network to learn different code representations for EMG windows originating from different 1-DoF motions:

$$\mathcal{L}_s = \max(0, 10 - \|\mathbf{C}_1 - \mathbf{C_2}\|_2^2) \tag{3}$$

The *divergence* loss $\mathcal{L}_d$ regularizes the ANN model by penalizing mixer output distributions $N(\boldsymbol{\mu}, \text{diag}(\boldsymbol{\sigma}))$ with large Kullback–Leibler divergence compared to the normal distribution with zero mean and unit variance:

$$\mathcal{L}_d = \text{KL}(N(\boldsymbol{\mu}, \boldsymbol{\sigma}), N(\mathbf{0}, \mathbf{1})) \tag{4}$$

Loss minimization was performed iteratively by use of the Adam algorithm [8] with $\eta = 10^{-4}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, training in mini-batches of size 1024 for a total of 5000 epochs. At the start of the training, all network parameters were given initial values via Glorot initialization.

D. Synthetic Data Evaluation

Signals from the held-out subject were split into classifier training and testing data on the basis of repetition: data from the first and second repetition of each motion for training and data from the last repetition for testing. This designation was used to produce 4 pattern recognition training datasets:

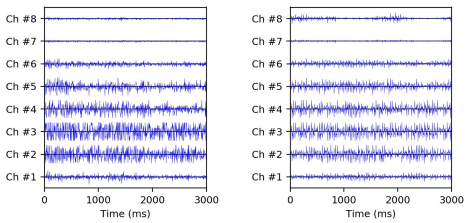
- **I: Complete**. Real EMG from all motions.
- **II: Pruned**. Real EMG from all 1-DoF motions.
- **III: Deep Augmentation**. Real EMG from 1-DoF motions appended with synthetic EMG from 2-DoF motions produced by the trained synthesizer model. This training set was created by feeding every combination of 1-DoF motion repetitions as inputs to the network; as the first two repetitions of each motion had been selected for pattern recognition training, this resulted in $2^2 = 4$ synthetic EMG repetitions per unique 2-DoF motion.
- **IV: Additive Augmentation**. Real EMG from all 1-DoF motions appended with synthetic EMG repetitions for all 2-DoF motions created by simply summing every possible pair of real EMG signals from 1-DoF motion repetitions. This reference training set was included as a comparison to ensure that any classifier performance gain brought about by the deep synthesizer model reflects properties of the synthesized signals and not simply an inflated training set.

For each of the 4 datasets, signals were segmented into feature vectors with a sliding window technique using 250 ms (50 samples) time windows with step size 5 ms (1 sample). A conventional time-domain feature set [9] (mean-absolute value (MAV), zero-crossings (ZC), slope-sign change (SSC), waveform length (WL)) was extracted from each window, producing a 32-dimensional feature vector at each window location. A target 2-dimensional ternary label vector associated with each feature vector was created by a DoF-wise majority vote over the samples of the time window. All 32 features were normalized to have zero mean and unit variance across each pattern recognition training set. A single pattern recognition test set was created in an identical manner using the last repetition of all 8 nonrest motions.

For each of the 4 pattern recognition training sets, one 3-class Linear Discriminant Analysis (LDA) classifier was trained per DoF; the $D = 2$ classifiers trained on each training set can together be viewed as a single multi-output, multi-class classifier whose performance is indicative of the quality of the training set. A choice of two multi-output classification metrics—Exact Match Rate (EMR) and Hamming Loss(HL)—were computed to quantify the performance of each of the 4 multi-output classifiers when operating on the test set. For both metrics, the difference in mean (across the 20 cross-validation iterations) was computed between the pruned (II) and deep augmentation (III) datasets, between the full (I) and deep augmentation (III) dataset, and between the deep augmentation (III) and additive augmentation (IV) datasets. Wilcoxon signed-rank tests with $\alpha = 0.05$ were used to assess the significance of observed differences.

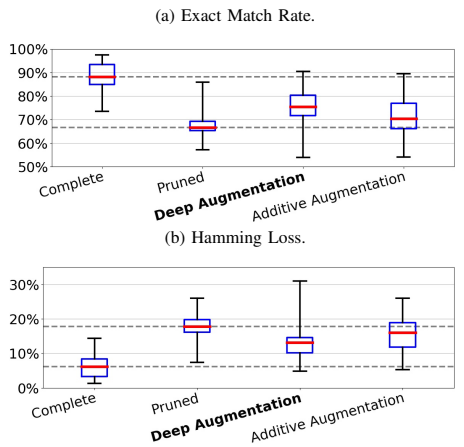
III. RESULTS

An example of a synthetic EMG window produced by a deep synthesizer model and an example of a corresponding real EMG window (i.e. from the same subject and motion) are shown together in fig. 2. Performance metrics achieved by LDA classifiers trained on complete (dataset I), pruned (dataset II), and partially synthetic (datasets III and IV) data are summarized in fig. 3. For the EMR metric, the mean increase between dataset II (67.86%) and III (75.42%) was 7.59% ($p = 0.00039$), the mean increase between dataset IV (70.60%) and dataset III was 4.82% ($p = 0.0064$), and the mean increase between dataset III and dataset I (87.96%) was 12.54% ($p = 0.000089$). Similarly for the HL metric, the mean decrease between dataset II (17.95%) and dataset III (13.41%) was 4.54% ($p = 0.00078$), the mean decrease between dataset IV (15.72%) and dataset III was 2.31% ($p = 0.014$), and the mean decrease between dataset III and dataset I (6.24%) was 7.17% ($p = 0.000088$).



(a) Real EMG window. (b) Synthetic EMG window.

Fig. 2: Example of real and synthetic EMG windows from the same 2-DoF motion and subject.



(a) Exact Match Rate.

(b) Hamming Loss.

Fig. 3: Multi-output performance metrics of LDA classifiers trained on datasets I-IV.

IV. CONCLUSION

This paper has proposed a deep learning-based approach for completing EMG pattern recognition training datasets containing only 1-DoF motions with synthetic EMG associated with multi-DoF motions. To assess the viability of the approach, the impact on classifier performance of partially synthesized data was investigated. The inclusion of EMG synthesized via the novel approach resulted in significantly higher performance (as measured by two metrics) compared to when using unaugmented data and augmented data created with a naive reference method. Nevertheless, the quality of the synthesized 2-DoF data was found to be strictly inferior to real EMG signals from 2-DoF motions for the purpose of training pattern recognition algorithms. Even so, these findings show that it is possible to model a user-independent relationship between EMG from multi-DoF motions and EMG from constituent 1-DoF motions, and that such models can be used to generate practically applicable training data.

Although this work only involved models operating on EMG from 2-DoF motions, there is no fundamental obstacle to extending the approach presented here to an arbitrary number of controllable DoFs. However, the need to record all combinations for the purpose of initially training a synthesizer model puts a practical upper limit on the number of DoFs manageable by the approach. In addition to extending the method to dataset with additional DoFs, future work could evaluate the use of alternative machine learning methods aimed at image or signal combining (e.g. [10]).

REFERENCES

[1] E. Scheme and K. Englehart, "Electromyogram pattern recognition for control of powered upper-limb prostheses: state of the art and challenges for clinical use," *J Rehabil Res Dev*, 2011, vol 48, pp. 443-59.

[2] A. Krasoulis and K. Nazarpour, "Myoelectric digit action decoding with multi-output, multi-class classification: an offline analysis," *Sci Rep*, vol. 10, no. 1, 2020.

[3] A. Furui, H. Hayashi, G. Nakamura, T. Chin, and T. Tsuji, "An artificial EMG generation model based on signal-dependent noise and related application to motion classification," *PLoS one*, vol. 12, no. 6, 2017.

[4] E. Campbell, J. A. D. Cameron, and E. Scheme, "Feasibility of data-driven EMG signal generation using a deep generative model," in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2020, pp. 3755-3758.

[5] A. E. Olsson, P. Sager, E. Andersson, A. Björkman, N. Malesevic, and C. Antfolk, "Extraction of multi-labelled movement information from the raw HD-EMG image with time-domain depth," *Sci Rep*, vol. 9, no. 1, 2019.

[6] A. E. Olsson, N. Malesevic, A. Björkman, and C. Antfolk, "Learning regularized representations of categorically labelled surface EMG enables simultaneous and proportional myoelectric control," *J Neuroeng Rehabil*, vol. 18, no. 35, 2021.

[7] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in *2nd International Conference on Learning Representations (ICLR2014)*, 2014.

[8] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization", Preprint available: http://arxiv.org/abs/1412.6980, 2014.

[9] B. Hudgins, P. Parker, and R. N. Scott, "A new strategy for multifunctional myoelectric control," *IEEE Trans Biomed Eng*, vol. 40, no. 1, pp. 82-94, 1993.

[10] L. A. Gatys, A. S. Ecker, and M. Bethge, "A neural algorithm of artistic style," Preprint available: https://arxiv.org/abs/1508.06576, 2016.

Författarens Tack

Den här avhandlingen blev i huvudsak till på grund av två grupper: de som aktivt bidrog till dess skrivande, och de som hade den goda smaken att låta bli. Min tacksamhet till båda är ungefär lika stor.

Först och störst i den första gruppen är min huvudhandledare **Christian Antfolk**. Jag vill rikta ett tack till dig för din ständigt öpna dörr som i kombination med ditt gränslösa förtroende gjort mina år som doktorand till de hittills mest lärorika i mitt liv. Mina biträdande handledare, **Nebojša Malešević** och **Anders Björkman**, är helt klart näst på tur. Nebojša, tack för dina raka svar och för eskapader i den serbiska elektrostimuleringens magiska värld. Anders, tack för dina granskande ögon och dina försök att få en enkel ingenjör att förstå att många saker är svårare än de kan verka. Ett stort tack är på sin plats till forna och nuvarande delar av neurotekniksgruppen: **Pamela**, **Jonathan**, **Robin** och **Jia**. Ert sällskap är alltid givande och er data nästan alltid användbar. I samma andetag är det på sin plats att tacka hela BME, i synnerhet **Ulrika** och **Ammi** som outtröttligt hanterat alltför sent inlämnade rekvisitioner och reseräkningar. Mina försökspersoner från BME och annorstädes förtjänar också ett tack för att de bidragit med sin tid och sin smärta.

Viktigast för mig i den andra gruppen är min käraste **Ina**. Tack för att du står ut med min skit och ständigt skänker mig glädje. Min närmaste familj, **mamma**, **pappa** och **Viktor**, har också spelat en roll i uppdraget att hålla mig välmående och någorlunda fungerande. Mina goda vänner **Johan**, **Melker**, **Lucas**, **Måns**, **Linus**, **Elliotte** m fl. ska också ha tack för att de ännu inte dödat mig, trots att hotet svävat i luften under många års tid. Att inte leverera ett stort tack till **Filip** är i det närmaste att betrakta som tjänstefel—utan din nit och redlighet som motvikt till min lättja hade jag sannolikt inte tagit någon examen alls, än mindre (möjlighet till) doktorsgrad.

Tack också till **GPT4** och **SciHub**, som var och en accelererat den här avhandlingens skrivande med en faktor ~ 1.2. Det största tacket av dem alla måste jag rikta till alla de tänkare som utfört det matematiska grovjobbet på vilket detta arbete vilar. Utan den uppsjö av häpnansväckande verktyg som redan fanns uppdukade vid projektets början så hade denna avhandling varit av en annan, långt mindre imponerande, form. Projektet finansierades av **Stiftelsen Promobilia**, som sjävklart också förtjänar ett stort tack.