On the application of hidden Markov models for signal decoding in the context of brain computer interfaces

Dissertation

zur Erlangung des akademischen Grades

Doktoringenieur

(Dr.-Ing.)

von Dipl.-Phys. Tim Pfeiffer

geb. am 24. November 1987 in Aschersleben

genehmigt durch die Fakultät für Elektrotechnik und Informationstechnik der Otto-von-Guericke-Universität Magdeburg.

Gutachter: Prof. Dr. rer. nat. Georg Rose
Prof. Dr.-Ing. Andreas Nürnberger
Prof. Dr.-Ing. Hermann Hinrichs

Promotionskolloquium am 29. November 2018

Zusammenfassung

Brain-Computer-Interfaces (BCI) stellen eine direkte Kommunikation zwischen dem menschlichen Gehirn und einem Computer her. Auf diese Weise ermöglichen sie die Steuerung von Assistenzgeräten, wie z. B. Rollstühlen oder Prothesen, ausschließlich auf der Basis von Hirnaktivität. Diese Technologie stellt einen enorm vielversprechenden Ansatz für die Unterstützung von Patienten mit schweren körperlichen Beeinträchtigungen dar, wie sie beispielsweise bei Querschnittslähmungen oder als Folge neurologischer Erkrankungen auftreten. Die zentralen Komponenten eines BCIs befassen sich dabei mit der Messung und Verarbeitung der Hirnsignale. Sie verfolgen das Ziel, die Intention des Nutzers aus dessen Hirnaktivität zu dekodieren um daraus ein Steuersignal für das gewünschte Assistenzsystem zu generieren. Die konkrete Zuordnung einer Intention aus den Daten erfolgt mit Hilfe sogenannter Klassifikatoren, die in verschiedenen Bereichen des maschinellen Lernens eingesetzt werden. Ein bekanntes Anwendungsbeispiel ist die automatische Spracherkennung (ASR), die im Zuge der Verbreitung von Sprachsteuerung in modernen elektronischen Geräten in den letzten Jahren weiter an Bedeutung gewonnen hat. Über die Jahre hat sich dort besonders ein Klassifikationsverfahren als Gold-Standard etabliert: die sogenannten *Hidden Markov Modelle* (HMM). Die exzellente Eignung von HMMs für die ASR begründet sich vor allem durch ihre Eigenschaft, zeitliche Variationen der Signale korrekt beschreiben zu können sowie ihren umfangreichen Möglichkeiten zur Einbeziehung von Vorwissen in die Erkennung—zwei Aspekte, die auch für die Dekodierung von Hirnsignalen große Vorteile versprechen. Dennoch finden sich in der Literatur bislang nur sehr wenige Studien, welche HMM-basierte Ansätze im Kontext der BCI-Signalverarbeitung verfolgten.

Diese Arbeit widmet sich einer ausführlichen Analyse von HMM-basierten Dekodierungsverfahren für unterschiedliche Fragestellungen aus dem Themenfeld BCI. Zunächst wird das Verfahren in sogenannten Single-Trial-Auswertungen evaluiert. Hierbei wird gezeigt, dass die erzielbaren Erkennungsraten von HMM-Klassifikatoren auf das Niveau einer Gold-Standard-Vergleichsmethode (Support-Vektor-Maschine) gebracht werden können und deren Ergebnisse in einzelnen Fällen sogar signifikant übertreffen. Als entscheidende Faktoren stellen sich neben der Wahl einer geeigneten Featureextraktion und -selektion vor allem die optimale Anpassung der gewählten Modellkonfiguration heraus. In einer weiteren Untersuchung wird analysiert, welche Vorteile sich aus den Eigenschaften von HMMs als dynamische Klassifikatoren ergeben. Hierfür wird eine Routine vorgestellt, mit deren Hilfe sich zusätzliche Informationen über die erkannten Ereignisse bei der Dekodierung ermitteln lassen, ohne dass ein separates Training der Routine erforderlich wird. Ergänzend zum Typ eines Ereignisses kann somit zeitgleich auch dessen Länge extrahiert werden kann.

Um ein realistischeres Einsatzszenario abzubilden, wird die Klassifikationsroutine für die kontinuierliche Erkennung erweitert. Im Rahmen dieser Arbeit wurde ein umfangreiches Software-Framework entwickelt, welches alle nötigen Anpassungen bereitstellt, um mit dem "Hidden Markov Model Toolkit" (HTK) eines der bekanntesten Softwarepakete aus der ASR auch für den Einsatz in BCIs nutzbar zu machen. Dem substanziell erhöhten Schwierigkeitsgrad bei der kontinuierlichen Erkennung wird durch die gezielte Einbeziehung von Vorwissen begegnet. Hierzu werden Informationen über häufig ausgeführte Aktionen genutzt, die sich durch geschickte Einbindung sogenannter Language-Modelle bei der Dekodierung berücksichtigen lassen. Als Ergebnis kann eine deutliche Reduktion der Fehlerraten bei der Klassifikation erzielt werden.

Zusammenfassend kommt diese Arbeit zu dem Schluss, dass HMMs eine äußerst vielversprechende Methode zur Dekodierung von Hirnsignalen im Kontext von BCIs darstellen. Es wird aufgezeigt, dass die Methodik, insbesondere durch die Möglichkeiten zur Vorwissensintegration, großes Potential für robuste, praktische Lösungen sowie weiterführende Entwicklungen bereithält.

Abstract

Brain-computer-interfaces (BCI) provide direct control over assistive devices using only brain activity. By that, BCI technology is a highly promising means for helping patients with severe handicaps, such as tetraplegia or paralysis due to neurological disorders. The crucial component in BCIs is processing of the acquired brain signals and identifying from the data the intention of the user. This is done with so-called classifiers, which are used in numerous topics of machine learning. A particularly prominent application example is automated speech recognition (ASR), which underwent a substantial increase in attention with the rise of voice control for electronic devices over the last years. For a long time, one specific classifier has been established as gold-standard for ASR: the so-called *hidden Markov models* (HMM). This dominance is mainly due to their ability to account for temporal variation within the signals as well as various possibilities to incorporate prior knowledge into the decoding—two features that promise great benefit also for decoding of brain signals. In the literature, however, only a small number of studies can be found that considered using HMMs in the context of BCI yet.

In this work, a comprehensive investigation of HMM-based decoding approaches is conducted in a variety of BCI settings. First, the performance of HMM classifiers is evaluated in so-called single trial decoding tasks. It is shown that decoding accuracies can be brought to the level of gold-standard routines (here: support vector machines) and in some cases, HMMs even outperformed the comparison method with significant performance advantages. This can be achieved by appropriate adaptions to the central components of the entire signal processing chain, most importantly by means of defining optimal model configurations, but also by finding suitable feature extraction and selection strategies. Furthermore, a method is developed to analyze model output (or more precisely, Viterbi paths), in order to assess the duration of an event without the requirement for a dedicated training routine. This approach takes advantage of the properties of HMMs that arise from the fact that they perform so-called dynamic decoding—a unique feature without equivalent in the widely-used static classifiers. To better reproduce typical application settings, the routines are extended to enable continuous decoding. For that purpose, the gold-standard speech decoding framework 'hidden Markov model toolkit' (HTK) is adapted for use in BCI decoding tasks and being embedded into an extensive software framework, developed particularly to allow for quick and convenient variation of all central decoding components. Continuous decoding substantially raises the difficulty of the setting. It is shown how this issue can be addressed by incorporation of *a priori* information into the decoding. By effective utilization of a common ASR technique, so-called language models, available prior knowledge is exploited, and substantial reduction in decoding errors is achieved.

In summary, this work provides evidence that the application of HMMs is a highly promising choice for decoding in the context of BCIs. It is shown that the routines—in particular with respect to the possibilities of prior knowledge incorporation—offer great potential for robust, practical solutions and future development.

Declaration of Honor

„I hereby declare that I produced this thesis without prohibited external assistance and that none other than the listed references and tools have been used. I did not make use of any commercial consultant concerning graduation. A third party did not receive any nonmonetary perquisites neither directly nor indirectly for activities which are connected with the contents of the presented thesis.

All sources of information are clearly marked, including my own publications.

In particular I have not consciously:

- Fabricated data or rejected undesired results
- Misused statistical methods with the aim of drawing other conclusions than those warranted by the available data
- Plagiarized data or publications
- Presented the results of other researchers in a distorted way

I do know that violations of copyright may lead to injunction and damage claims of the author and also to prosecution by the law enforcement authorities. I hereby agree that the thesis may need to be reviewed with an electronic data processing for plagiarism.

This work has not yet been submitted as a doctoral thesis in the same or a similar form in Germany or in any other country. It has not yet been published as a whole."

Magdeburg, December 10, 2018

Tim Pfeiffer

Acknowledgments

First and foremost, I want to thank my wife Patricia for her continuous support and her understanding when I spent more hours in the office than at home during the final stage of this work. I love you.

Many thanks are due to my supervisor Professor Georg Rose, who gave my the opportunity to work on this fascinating topic, not least because he always managed to find a funding for my position throughout the years. I am truly grateful for the trust placed in me.

Discussing results and hypotheses is fundamental to any scientific investigation. This is why helpful friends and colleagues play such an important role and I want to thank everyone who supported me in that respect. In particular, I thank Nicolai Heinze and Robert Frysch for the great teamwork and for sharing ideas on the various topics. Big thanks are also due to Jaspar Pahl, who made substantial contribution to my work by fighting that evil demon, called “HTK toolkit,” with all its frustrating (or non-existent) error reports—so, what exactly was `ERROR [+2128]` again?

I want to thank Professor Robert T. Knight for giving me the opportunity to visit his lab and work with some of the rarest ECoG data, as well as for all his constructive comments on various publication manuscripts.

This work would not have been possible if it were not for all the people involved in the data recording. In particular, I want to express my gratitude to all patients that volunteered for BCI studies; without them, BCI research would most certainly not be where it is today. In that respect, I also thank the entire team from the ECoG recording sites in Stanford and San Francisco. Furthermore, I want to thank the MEG team from Magdeburg, especially Stefan Knape for a great introduction to the acquisition setup and his immediate support whenever we came across any technical problems. I also thank Felicitas Detmer for spending numerous hours in the MEG lab and recording data from our volunteering subjects (My thanks to all of you too!).

Last but not least, I wish to thank my family. You supported me in every possible way ever since I can remember; and without some occasional remarks from your side that it is about time for me to finally finish my PhD, who knows, this thesis might still be undone.

Funding

This work is partly funded by the German Ministry of Education and Research (BMBF) within the Forschungscampus *STIMULATE* (grants 03FO16102A and 13GW0095A).

Contents

List of Figures

Nomenclature

ASR	automated speech recognition
BCI	brain-computer-interface
BMI	brain-machine-interface
CM	confusion matrix
CV	cross-validation
DB	Davies–Bouldin
ECoG	electrocorticography
EEG	electroencephalography
HG	high gamma
HMM	hidden Markov model
HTK	Hidden Markov Model Toolkit
LFTD	low-frequency time domain
MEG	magnetoencephalography
MRalt	movement–rest alternation
MSI	multi-stimulus-issue
NIRS	near-infrared-spectroscopy
PK	prior knowledge
SB	sub-block
SC	semi-continuous
SNR	signal-to-noise ratio
SVM	support vector machine

1. Introduction

What is a brain-computer-interface? A brain-computer-interface (BCI, often also brain-machine-interface (BMI)) is a device that analyses brain signals to extract command signals for external applications, intended to assist people in putting their intentions into practice [1]. A common definition describes a BCI as "a communication system that does not depend on the brain's normal output pathways of peripheral nerves and muscles" [2]. By that, it replaces the functionality of nerves and muscles as well as their induced movements by hard- and software components that work based on the underlying electrophysiological signals [1]. To realize such a system, three central components are required: data acquisition, signal processing and application implementation (Figure 1.1). A variety of approaches is used to acquire brain signals from human subjects. Almost all techniques measure electric or magnetic field components that go along with neural activity in the brain. The most prominent means to pick up the electric field is electroencephalography (EEG), invented by Hans Berger back in 1929 [3]. The acquired brain signals are then analyzed using a broad variety of signal processing and machine learning techniques with the goal to extract the user's intention. The specific type of this intention (e.g. movement, speech) varies between the actual application it is meant to control. Successfully extracted command signals can be used to drive the actuator—the device that translates the user intention into practice—which is realized either by means of hardware (e. g. wheelchairs or robotic prostheses) or software (e. g. communication or spelling devices) components.

Applications (Who needs BCIs?) The BCI technology is primarily intended to support disabled patients with everyday-life activities in order to increase their level of autonomy. An important part of the patient collective consists of persons with motorical disabilities, be it due to tetraplegia, loss of limbs (e. g. amputees or accident victims) or paralysis as a consequence of stroke. The goal for these patients is restoration of mobility. To achieve this goal, BCIs might be used to steer wheelchairs or even grant control over robotic prostheses. Another target group are patients with severe neurological disorders (e. g. locked-in syndrome, amyotrophic lateral sclerosis, stroke) that have lost the ability to communicate. In this case, BCI systems aim at providing a means of communication that relies only on brain activity.

Besides restoration of mobility or ability to communicate, BCIs are also used for so-called *neuro-feedback* purposes. The idea behind this is to monitor brain activity during the performance of specific (mental) exercises and presenting to the user a feedback—usually provided through a visual channel, e. g. a computer screen—about how well the task is performed. This feedback is computed from the measured brain activity and can display a variety of neuro-physiological parameters like level of mental focus or relaxedness of the

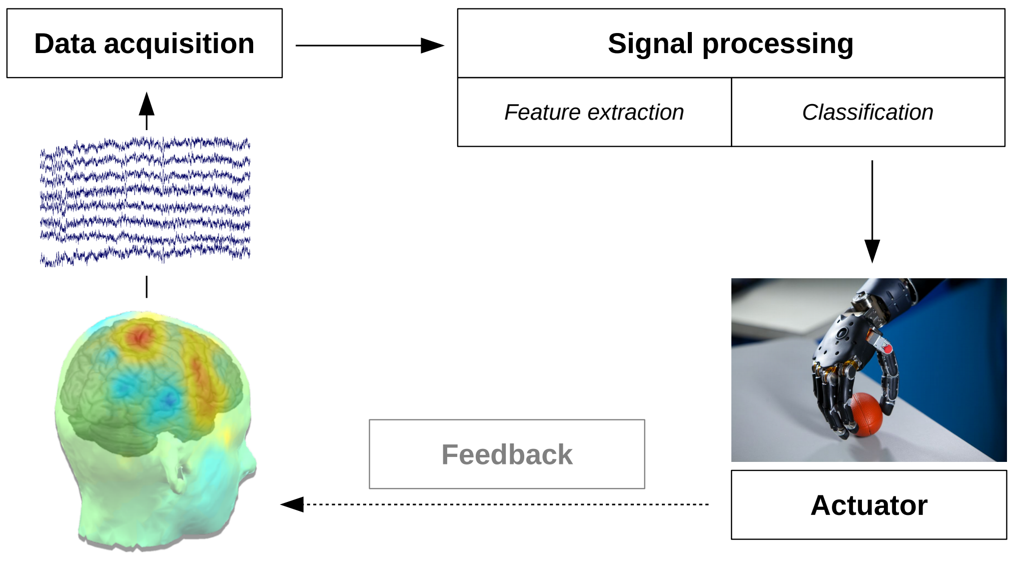


Figure 1.1: Schematic overview of the structure of a brain-computer-interface. A BCI consists of three components. First, brain signals are acquired using appropriate acquisition techniques. Signal processing methods are used to extract essential information from the recorded data and decode the user's intention. Based on this prediction, a command is sent to an actuator (e. g. a robotic prosthesis) that translates the intention into practice.

subject. Due to the immediate feedback information, subjects are able to learn how to self-regulate their mental state. Such systems are used to improve therapy of neurological disorders, most prominently attention deficit hyperactivity disorder (ADHD) [4]. Similar approaches can also be used to augment rehabilitation therapy for stroke patients [5]. Using movement intentions decoded from the recorded brain signals to trigger the assistance with the execution of the actual movement, a close coupling between neural activity and perceived motion feedback is achieved. It has been shown that this supports regeneration by increasing the effect of neural plasticity compared to conventional passive movement therapy. Furthermore, BCI technology can also be used for patient home monitoring [6] (e. g. epilepsy seizure prediction [7]). In addition to medical applications, BCI is also an emerging trend in entertainment industry (esp. in video games [8, 9]).

BCI development overview The concept of BCI first appeared in the literature in a paper from 1973 by Jacques Vidal [10]. At that time, possibilities for BCI solutions were severely constrained by the limited computational power and lack of memory resources of the available computer hardware. Due to the rapid development in this sector, BCI technology became more feasible, so that by the end of the 1990s, first real BCI applications could be implemented. In 1999, the group of Birbaumer et al. demonstrated a spelling device that allowed patients suffering from the locked-in syndrome to communicate based on interpreted brain signals [11]. Until the year of 2004, all documented BCI systems for human subjects were based on non-invasively recorded signals (esp. EEG). Leuthardt et al. were the first to present an approach based on electrocorticography[1] (ECoG) signals in human subjects [12]. The recorded signals were used to provide the user control over a one-dimensional cursor. In this study, a significant advantage of ECoG data over non-invasive recordings is taken advantage of—the substantially extended range of usable frequencies. Whereas typical non-invasive brain signal recordings cannot provide useful information for frequencies components above approximately 40 Hz, ECoG opens up the possibility to utilize signals all the way up to 200 Hz and more. This promoted extensive investigation of higher frequency components in the recorded brain signals. Especially the so-called high gamma band turned out to yield particularly high information content. The properties of high gamma signals were documented elaborately in the work of Miller et al. [13–16] and used in a broad variety of scenarios since then. Their favorable signal-to-noise ratio and high spatial resolution allowed for much more complex applications. Invasive BCI-systems have been used to predict movements of individual fingers [17, 18], decode speech [19, 20], and even to control robotic arm prostheses in paralyzed patients [21–23].

When it comes to decoding the brain signals, a broad variety of algorithms have been studied. These cover different kinds of neural networks [24–26], decision trees [27, 28] and support vector machines (SVM) [29–31]—just to name a few. Interestingly, most of these approaches are so-called *static classifiers*, which disregard the temporal development of the measured signals [32]. This is surprising considering the dynamic nature of brain signals. In other fields of machine learning applications that deal with comparable data, *dynamic classifiers*[2] have established as gold-standard. This holds true particularly when it comes to automated speech recognition (ASR). The broad variance found in articulation, even for one and the same speaker, makes it necessary to use decoding approaches that consider this circumstance appropriately. *Hidden Markov models* (HMM) are used in almost all state-of-the-art ASR systems to tackle this problem [33, 34]. In BCI research however, HMMs have not been used that frequently to decode brain signals yet.

Related work As mentioned before, the amount of studies in which HMMs have been used to decode brain signals in the context of BCIs is rather limited. Table 1.1 shows an overview of studies with HMM-based decoders used for BCI classification tasks with human subjects.

As one of the first, Obermaier et al. applied an HMM decoder for discrimination between motor imagery of left and right hand movements from human EEG recordings in an offline setting [35]. The approach has later been extended to an online setting [36]. Comparing their results to a decoder based on linear discriminant analysis, they reported superiority of the HMM classifier in 11 out of 12 cases. The advantage was particularly substantial for online use, which the authors attribute to the better generalization of HMM decoding. The group of Sitaram et al. [37] performed a similar experiment using near-infrared-spectroscopy

[1]Electrocorticography—sometimes also referred to as intracranial EEG—is an invasive recording technique in which brain signals are recorded directly from the surface of the cortex. A more detailed description of the method can be found in Section 2.1.1.

[2]A detailed description of the differences between static and dynamic classifiers can be found in Section 2.3.1.

Why to use HMMs for BCI?

Typical speech can be represented by a collection of about 30 phonemes and individual words are constructed as well-defined concatenations of these phonemes. On the next higher level of complexity, these words build the basis for a language that follows certain grammar rules and allow for context sensitive interpretation. Speech underlies significant variances when it comes to actual human speakers. Subject-individual pronunciation of words, personal preferences towards specific phrasing and several other aspects increase complexity and complicate automated recognition. Despite all those difficulties, modern ASR systems provide highly accurate decoding results with a minimum of required training. The key ingredient to make this possible is the incorporation of prior knowledge into the recognition routines. By concept, probabilistic classification routines are heavily favored when it comes to convenient integration of additional knowledge in comparison to their non-probabilistic counterpart.
The challenges in many BCI-related problems are largely similar to typical difficulties that must by dealt with in ASR solutions. People's everyday life actions are composed of individual submovements that are concatenated to form meaningful sequences depending on the context. This concept has close resemblance to the structure of speech and language. Just like speech, brain signals have dynamic nature and their appearance may not only vary substantially between individuals but also even between different environmental conditions. As mentioned before, HMMs are the primary candidate of choice in the field of speech recognition, in which one faces closely similar issues. This suggests that HMMs potentially also provide a highly suitable candidate for brain signal decoding and BCI applications.
In summary, compared to most gold-standard BCI classifiers, HMMs primarily offer two major advantages:

1. Probabilistic classification,
2. Dynamic decoding with time-warping properties.

(NIRS, cf. Section 2.1.3). They also documented an advantage of HMM decoders, in this case over support vector machines, which represent one of the gold-standard classifiers in BCI (cf. Section 2.3.3). Chiappa and Bengio [38] investigated the advantage of using dynamical models. To do so, they compared HMM decoders with a static version (i. e. with a single state, cf. Section 2.3.2) in a three-class decoding problem consisting of two types of motor imagery (same as in the previously mentioned studies) and mental generation of words. In their results, no significant advantage could be found for the dynamic approach. Still, the authors showed that the HMM-based decoder achieves decoding accuracies significantly above chance level even when trained on data that has been recorded several days earlier. This is an important finding for potential real-life applicability of such a system. The study by Lederman and Tabrikian [39] reports an interesting approach of using HMMs as an ensemble classifier. For signals from each of the individual EEG electrodes, they trained a separate pair of HMMs (i. e. one model for each class) and selected from the entire set of models the one that fitted best for the test data. They tested their routine on two publicly available EEG datasets and reported significant improvements with the HMM approach compared to the results of various other groups.

A detailed comparison of various decoding algorithms has been published by Rezaei et al.

[40]. The authors compare the performance of five common classifiers (including HMMs) in discriminating five different mental tasks from human EEG recordings. Classification has not been carried out for the full the five-class problem, but for the simplified case of binary "duels" between all possible combinations of tasks (10 combinations in total). The reported decoding accuracies were lowest for the HMM classifier, with substantially lower performance than the other four routines (HMM: ~65 %, others: ~86–90 % correct classifications).[3] Similar findings have been reported in Cincotti et al. [42] in a comparison of three classifiers (including HMM) for left and right hand motor imagery. In the work of Zhao et al., HMMs have been used to decode joystick movements from human ECoG recordings [41]. The quite challenging experimental task was to discriminate between eight different target locations to which the subject had moved a cursor with a joystick. Potentially due to the difficulty of the task, decoding accuracies reached only about 20 %—a result that is just slightly above chance level (12.5 % for an eight-class problem). Nevertheless, the group documented an interesting extension of coupling two HMMs into a single decoder, and that work is one of the only studies demonstrating the use of HMM decoders for ECoG data.

In summary, several studies report promising results for the use of HMM classifiers in BCI-context. However, there are also contradictory findings. Consequently, there is the demand for further analysis. Three aspects are of particular interest: First, the majority of studies focused offline or online single trial analysis (cf. Section 2.3.1). Continuous decoding and prior knowledge incorporation are rarely found. This is an interesting observation, as exactly for those features, HMMs have established as gold-standard in ASR. Second, HMMs have been used almost exclusively for analysis of non-invasive data, with a distinct focus on EEG recordings. With the exception of [41], none of the listed studies is based on ECoG data. And finally, there is a strong dominance of motor imagery based experiments with only few classes and rather long trial duration.

Much more frequently than for actual decoding, HMMs have been used as an additional processing layer in BCI-based spelling devices. These systems are usually based on P300 [47] or steady-state-visually-evoked-potential (SSVEP) [48] paradigms that allow users to sequentially select individual characters based on visual attention. Different algorithms are applied to detect which of the letters is attended by the subject. In several approaches, HMMs are then used to incorporate prior knowledge (PK) on the likelihood of spelled words or phrases by means of language models. Almost all those studies report significant performance improvements with the use of such techniques. Comprehensive reviews of the use of language modeling in BCI speller applications are provided by Mora-Cortes et al. [49] and Speier et al. [50].

Another interesting approach has been shown by Moses et al. [20]. The group uses a phoneme-based Viterbi decoder—a method similar to single state HMMs—to decode speech directly from neural signals (ECoG). In their study, the authors apply common ASR techniques (i. e. n-grams, cf. Section 2.3.4) to improve the accuracy of their decoder, again demonstrating the importance of PK incorporation for BCI applications.

When it comes to motor BCI tasks, the focus of interest more and more shifts towards PK

[3] Unfortunately, the authors do not provide a comprehensible interpretation of the poor HMM performance ("This can be attributed to the special implementation of HMM in this work.", [41]).

Table 1.1: Examples of BCI studies using HMM classification. K denotes the number of classes in the investigated decoding problem and T the duration of individual trials.

Study	Year	Task	Modality	K	T
Obermaier [36]	2001	left/right hand imagination	EEG	2	5 s
Obermaier [43]	2001	5 mental tasks	EEG	2-5	4 s
Cincotti* [42]	2003	left/right hand imagination	EEG	2	N/A
Chiappa* [38]	2004	left/right hand imagination & mental generation of words	EEG	3	1-3 s
Rezaei [40]	2006	5 mental tasks (10 binary problems)	EEG	2	10 s
Sitaram [37]	2007	left/right hand imagination	NIRS	2	10 s
		left/right hand finger tapping	NIRS	2	10 s
Lederman [39]	2012	left/right hand imagination	EEG	2	6 s
		hand up-/downward imagination		2	5.5 s
Zhao* [41]	2014	Joystick movements (BCI2000 CursorTask protocol)	ECoG	8	1 s
Wissel [44]	2013	finger tapping (4 fingers)	ECoG	4	~ 0.5 s
Pfeiffer [45]	2016	picture category decoding	ECoG/MEG	3	2.1 s
Pfeiffer [46]	2018	finger tapping (4 fingers + rest)	ECoG	4+1	cont.

* conference paper (no journal article available), cont. – continuous decoding

incorporation. Some studies reported promising results in using PK to improve the accuracy of asynchronous[4] prediction of finger movements. The group of Wang et al. [51] considered biomechanical constraints of finger flexion to reduce the degree of freedom for decoding of such movements. By incorporating these constraints into their switching non-linear dynamic system that is used for decoding, significant increases in decoding precision could be achieved. Based on the same dataset, Flamary et al. [52] proposed an algorithm that uses the information about which finger is moving to make predictions on the movements of the other fingers. They could show that—with appropriate tuning of the algorithm—high correlation between real and predicted movements can be achieved. In the study from Delgado Saa et al. [53], typical change rates between movement and rest episodes are considered to produce more realistic decoding output. Recent studies further apply techniques of PK incorporation for reconstruction of movement trajectories from non-human primate ECoG data (e. g. [54, 55]).

Although a variety of approaches documented the incorporation of PK in the context of motor BCI scenarios, straightforward application of ASR (language models) methods cannot be found. As a general observation, it appears that there exists a large gap between simple approaches that are limited to select between different options and the attempt to

[4]In this context, 'asynchronous' means that the decoder must identify when events happen in addition to determining their type. More details on the difference between synchronous and asynchronous setting are provided in Section 2.3.1.

exactly predict entire movement trajectories with several degrees of freedom. The approach of using PK incorporation to construct meaningful combinations from decoded elementary components—a method that is already commonly used in spelling devices—is not investigated thoroughly for motor BCI tasks.

Objective and structure With the aim of providing a comprehensive investigation of the use of HMM classifiers in a variety of decoding scenarios, this work addresses the following main objectives:

1. Evaluation of hidden Markov model decoders in single trial decoding problems,
2. Investigation of potential benefits of dynamic decoding for BCI-related tasks,
3. Demonstration of continuous brain signal decoding with HMM-based decoders, and
4. Incorporation of prior knowledge with HMM-based decoders.

The remainder of this thesis is structured as follows. Chapter 2 provides the essential theoretical background. In Chapter 3, all methods used for the investigations are introduced. Additionally, relevant information on the experimental paradigms and the acquired data are presented. As the central part, Chapter 4 contains all results along with the associated discussion thereof. The main discussion Chapter 5 reviews the results in a broader context. This includes comparison of the findings to related work as well as analysis of the limitations. In the final content Chapter 6, potential future development is outlined. The thesis closes with a list of references and an appendix with supplementary material.

2. Theoretical Background

2.1. Data acquisition modalities

Signal transmission within the human brain is performed with the help of specialized cells, so-called neurons. These cells typically consist of a cell body with membrane extensions for excitation input (dendrites) and output pathways (axon). Electrical signals are transmitted along the cell—directed from the dendrite to the axon—by changes in the membrane potential that are generated by in- and outflow of ions into or out of the intracellular space. Ion flow rates are regulated by ion channels located in the cell membrane. Opening and closing (so-called gating) of these channels is mediated either based on voltage or on the presence of certain substances (ligands). Voltage-dependent channels play the key role in the generation of action potentials which represent the main mechanism of excitation transport. The excitation signal is transported along the axon and transmitted to other neurons at interconnections called synapses. Usually, these synapses connect the axon of the emitting neuron with the dendrite of the receiver. The majority of neurons also has a large number of synapses that collect signals from various inputs. From the dendrite, incoming signals are transported along the neuron towards the axon hillock—the part of the neuron's cell body that connects to the axon—where all input is summated. If incoming signals accumulate to a sufficiently high depolarization of the axon hillock for the voltage-dependent channels to open, an action potential will be generated and propagate along the axon.

The transmission of a signal by a neuron generates an electromagnetic field that extends to the space outside the neuronal cell itself. If larger collectives of neurons are aligned with similar spatial orientation and fire in a more or less coherent pattern, the individual field contributions superpose constructively and result in potentials that can be picked up on the human scalp using appropriate measurement techniques. The most common method to record the electric component of the electromagnetic field is *electroencephalography* (EEG)[1]. As electrodes are placed on the surface of the scalp, distance between signal origin (i. e. cortical neurons) and the sensor is quite substantial, with typical values in the order of 10–20 mm [59,60]. This distance induces spacial blurring of the electric field and as a consequence, signal quality deteriorates. Higher signal quality can be achieved with invasive recordings such as *electrocorticography* (ECoG) or intracranial electrodes. Like every other electric current, neuronal activity also induces a magnetic field. The magnetic field distribution is less sensitive to disturbances by matter and hence, constitutes an interesting alternative for sensing information on the signal generating neural processes. However, measurement of this field component requires very sensitive instrumentation as the field

[1]Comprehensive reviews on EEG can be found e. g. in the articles [56, 57] or in the book by Niedermeyer and da Silva [58].

strengths to be recorded are extremely low. The corresponding acquisition modality is called *magnetoencephalography* (MEG) [61].

Investigations in this work are based on ECoG and MEG data. These two modalities are briefly introduced in the following.

2.1.1. Electrocorticography

Electrocorticography (ECoG) is an invasive method to record brain signals directly from the surface of the cortex. The procedure requires opening of the patient's skull (craniotomy) in order to place a grid of electrodes on the brain. Electrodes can be placed either on top of the dura mater (epidural recording) or below it (subdural recording). Subdurally placed electrodes are closer to the signal source and thus achieve higher spatial resolution [62]. This method accounts for the majority of ECoG implantations. However, as subdural placement involves opening of the dura mater, it goes along with additional risk of inflammation. After the electrode grid has been put in place, the skull is reclosed. ECoG recordings are used as a tool to localize epileptic foci in patients with intractable epilepsy—i. e. forms of the disease that do not respond to conventional treatment with medication—in preparation of surgical treatment. The measurement is also used to determine whether the affected regions of the brain are related to vital functions (e. g. speech). If this is not the case, resection of epileptogenic brain tissue is a viable treatment option. Typical ECoG recordings are carried out over a period of several days during which some patients voluntarily participate in BCI studies[2]. After successful identification of epileptic foci, or if no conclusive indication can be found within reasonable time, electrode grids are explanted in an additional surgery.

Concerning data quality, ECoG signals have strong advantages over non-invasively recorded EEG data. ECoG provides spatial resolutions of up to 2.5 mm (subdural) [62] with high signal amplitudes of about 50–100 µV (compared to EEG: 10–20 µV) [12]. The excellent time resolution of ECoG recordings allows for meaningful analysis of signals up to 500 Hz. Commonly, EEG recordings do not contain anymore useful information for signal components above 40 Hz. The main reason for these differences lays in the electrical capacities and conductivities of the tissue in between the signal source and the measurement location. These properties cause a low-pass filtering effect for the signal [1]. Since source-to-detector distance is much smaller in ECoG recordings, the thickness of this in-between tissue is reduced to a minimum and consequently, low-pass filtering appears to a much lower extent. Additionally, high frequency signals are typically generated by small cortical structures (referring to their spatial extent) and are therefore found more prominently in electrodes close to their origin [1]. Besides providing higher signal quality, ECoG is also less susceptible to typical artifacts found in EEG or MEG recordings [12]. This refers mainly to artifacts caused by eye movements (EOG) or other muscle activity (EMG), for instance from face or neck musculature. The insensitivity to these types of artifacts makes ECoG a promising means for real-life BCIs as those require usage under conditions that naturally involve muscle activity and eye blinking.

[2]Except for the very rare case of patients (only a handful worldwide) who are solely volunteering for BCI projects—i. e. there is no clinical need for ECoG implantation in these patients—epilepsy patients are the only possibility to acquire invasively recorded data for BCI studies.

2.1.2. Magnetoencephalography

A major drawback of EEG measurements is the poor spatio-temporal resolution. Magnetoencephalography (MEG) presents a compromise solution that trades off the mobility of an EEG setup for higher signal quality. Contrary to the electric field, magnetic fields are not disturbed to the same extent by tissue and bone. Hence, measurement of the magnetic field distribution on scalp level allows for much more precise inference of the underlying signal sources than what can be achieved based on the corresponding electric field [63]. Therefore, information on activity of spatially close cortical structures can be assessed with high accuracy [64]. Typically, MEG recordings provide spatial resolution of up to 2–3 mm [65]. This can become crucial for a variety of BCI solutions that rely on information from such areas. A prominent example is the motor BCI scenario, as typical motorical areas of the brain (e. g. hand movement control) represent complex multi-dimensional information on a highly confined space (typically no more than a few square centimeters). To achieve these high spatial resolutions (0.5–2 mm [64]), a technique called magnetic source imaging is used [66]. Its goal is the reconstruction of the signal sources (i. e. neural activity) from the measured surface signals. This inverse problem is very challenging and has been addressed by a variety of approaches (e. g. equivalent current dipole models, beamforming). In addition to their high spatial resolution, MEG signals also provide good temporal resolution of less than 1 ms [65].

The magnetic field strengths to be measured are of extremely low amplitude. Typical signals reach about 10 fT to 1 pT, which is roughly 8–9 orders of magnitude lower than the earth magnetic field ($\sim 50\,\mu\mathrm{T}$). This circumstance raises high demands on the acquisition setup. Such low field strength can only be measured with so-called superconducting quantum interference devices (SQUID). This technique—introduced in the late 1960s by James Zimmerman—is based on a quantum mechanical effect of flux quantization to achieve the required level of sensitivity. SQUID sensors operate under extremely low temperatures that are achieved with liquid helium cooling. This requires significant technical effort, especially given the demand to have the sensors in close proximity to the head of a subject. Recordings are performed in electromagnetically shielded rooms to minimize effect of external noise sources. Furthermore, special attention must be paid regarding all additional equipment that shall be used. In particular, this means that no magnetic materials, moving metallic (or otherwise conductive) objects and electric devices (e. g. electric engines) might be used within the room as they would potentially disturb the recordings.

In summary, MEG is a means to assess brain activity in a fully non-invasive way that provides good signal quality (esp. with respect to spatio-temporal resolution) in comparison to other non-invasive techniques. Therefore, MEG comes with the advantage of non-invasive acquisition to be able to gather data from larger groups of subjects while still supplying sufficiently high data quality to study complex BCI-related problems. The high technical effort and the immobility of the acquisition setup severely limits the usability of MEG for use in everyday life BCIs; nevertheless, it is a highly useful tool for basic research and offers great potential for clinical diagnostics or neural rehabilitation applications.

Table 2.1: Comparison of different acquisition modalities for brain signal recording (from [67]).

Method	Activity	Measurement	Resolution		Risk	Portability
	measured	(Dir./Indir.)	temporal	spatial		
EEG	Electrical	Direct	~0.05s	~10mm	Non-inv.	Portable
MEG	Magnetic	Direct	~0.05s	~5mm	Non-inv.	Non-portable
ECoG	Electrical	Direct	~0.003s	~1mm	Invasive	Portable
Intracortical neuron recording	Electrical	Direct	~0.003s	~0.5mm (LFP) ~0.1mm (MUA) ~0.05mm (SUA)	Invasive	Portable
fMRI	Metabolic	Indirect	~1s	~1mm	Non-inv.	Non-portable
NIRS	Metabolic	Indirect	~1s	~5mm	Non-inv.	Portable

Abbr.: LFP–local field potentials, MUA–multi-unit activity, SUA–single-unit activity

2.1.3. Other modalities

Besides ECoG, intracranial electrodes represent another invasive technique that is used quite commonly. From all acquisition modalities, intracranial electrodes provide the highest spatio-temporal resolution, allowing for measurement of signal spike activity down to the level of individual neurons. These advantages go along with a series of downsides. Since the electrodes penetrate cortex tissue, this method is even more invasive than ECoG. Besides the increased risk of inflammation and permanent tissue damage, intracranial electrode also have a tendency to be encapsulated after a certain period of time, which significantly diminishes data quality.

As an alternative to measurement of electric or magnetic fields that go along with neural signals, oxygen consumption in the brain can be used as an indicator for neural activity. This is used in functional near-infrared spectroscopy (fNIRS) and blood-oxygen-level dependent (BOLD)-contrast imaging in functional magnetic resonance imaging (fMRI). While both methods are fully non-invasive, only fNIRS can be considered a suitable candidate for real-life BCI applications due to the immobility of MRI setups. However, similar to MEG, basic research using fMRI can provide useful insight into neural processes and may help to understand specific mechanisms that are involved in typical BCI tasks (e. g. motor imagery). A comparison of all modalities is shown in Table 2.1.

2.2. Signal processing

Usually, brain signals are acquired with a setup that contains multiple sensors (e. g. electrodes, see Section 2.1). These record neural activity from various spatial locations. For most cases, only a fraction of the acquired information is relevant for the specific decoding problem. Appropriate signal processing routines play a crucial role in facilitating meaningful decoding of the signals. The two essential steps in the processing chain are *feature extraction* and *feature selection.*

In combination, both steps have the goal of dramatic reduction in data dimensionality without loosing the desired information. The following example illustrates the necessity to compress information in a hypothetical BCI scenario. Assuming that the BCI is based on brain signal data acquired from 100 sensors with a sampling rate of 1000 Hz (typical values for BCIs) and task-relevant neural processes are expected to be found in a two second segment of data, this sums up to a total of

$$100 \cdot 2\,\mathrm{s} \cdot 1\,000\,\mathrm{Hz} = 200\,000$$

values that are recorded in that setup. In this example, the desired application shall be control of a wheelchair by triggering one of the following five commands: forward, backward, left, right, or stop. Hence, only 3 bits of information ($2^3 = 8$ possible values > 5 required commands) are sufficient to encode all the possibilities. Under the assumption that the measured signals are digitized with 16-bit precision, this results in

$$200\,000\,\mathrm{samples} \cdot 16\,\frac{\mathrm{bit}}{\mathrm{sample}} = 3\,200\,000\,\mathrm{bit}$$

of acquired data. Consequently, six orders of magnitude lay between the amount of measured data and the actually required command information. Without appropriate dimensionality reduction, it is outside the capabilities—even of powerful machine learning algorithms—to robustly assign the correct command to such a data segment. This problem is further compounded by the small amount of data that is available in typical BCI studies.

2.2.1. Feature extraction

Feature extraction routines play the central role in reducing the typically high dimensional input data to the (task-)relevant information. This is done by extracting the essential data properties that are of interest for a given problem. These techniques are broadly used in a variety of application areas, such as image (and video) processing or speech recognition as well as many other machine learning and pattern recognition topics. In case of brain signal decoding, feature extraction mainly pursues the goal of filtering out neural signals that are uncorrelated to the task under consideration and separating from them all task-related activity. A broad variety of techniques have been applied in different contexts [68]. One possible way of structuring these routines is to separate between three categories: *temporal*, *spectral* and *spatial features*. Some exemplary routines from all three types are briefly introduced in the following.

Temporal features Temporal features are processed versions of the recorded raw data in the time domain. These features are suitable to identify phase-locked information in the data. This refers to typical signal deflections—an example being prominent amplitude peaks as a response to external stimulation, like the so-called P300—occurring reproducibly at (almost) identical instants in the time series. To emphasize these signal characteristics, several processing techniques can be applied. Usually, phase-locked signal characteristics are restricted to low frequency components. Therefore, the majority of approaches focuses on low-pass filtering with subsequent downsampling of the time series to reduce data dimensionality. Among the most prominent methods to extract temporal features are finite impulse response filter (FIR) [69], Wiener filter [70], Wavelet transform [71], as well as other methods like template matching and autoregressive modeling approaches [72].

Spectral features Contrary to temporal features, with spectral features one tries to extract time-locked information, irrespective of signal phase. To do so, a common strategy is computation of power spectral densities, representing the frequency composition of the signal. This is often performed by means of Fourier transform [73] or, alternatively, using band-pass filtering with subsequent computation of signal envelopes (e. g. based on Hilbert transform [74]). Additionally, wavelet transforms can also be used to assess spectral information [75].

While these approaches can in general be applied to all frequency bands, their most important application is the analysis of high frequency components—often referred to as asynchronous or broadband activity. This activity has been found to encode a variety of different phenomenons which are of strong interest for potential BCI applications. The majority of studies focuses on information from the so-called *high gamma* (HG) band. Definitions vary slightly with respect to the exact frequency range that constitutes the HG band. Usually, frequencies in the range of 60–200 Hz (and above) are considered [76–78]. In the context of BCI, HG features have been successfully used to decode motor activity [14, 16, 79], for speech reconstruction [19, 20, 80], and in visual perception tasks [81–83]—to name just a few examples.

Spatial features As an extension of the methods described in the previous two paragraphs, spatial features also take into consideration the spatial distribution of the signals on the brain. This can be particularly helpful if signals exhibit characteristic distribution patterns that allow for distinction between different classes. Given that most non-invasive recording setups provide simultaneous data acquisition of large portions of the brain, spatial features are a particularly promising analysis tool for these type of signals. The most frequently used routine is called common spatial patterns (CSP). CSP is a supervised (data-driven) statistical learning approach that aims at determining a spatial filter that maximizes the variance of the signal of one class while at the same time minimizes the variance for the other class [84]. It was originally designed for discrimination between two different classes. However, due to its success in EEG analysis, multi-class extensions have been developed [85]. Spatial features are less commonly used in the analysis of invasively recorded data since data is usually acquired from a limited area only (e. g. typical ECoG grid dimensions of $8\,\text{cm} \times 8\,\text{cm}$).

2.2.2. Feature selection

Feature selection routines are meant to determine the most informative features with respect to the investigated task. As a result, a reduced subset of features is constructed that should ideally contain all relevant information required for precise decoding while reducing data dimensionality as far as possible.

In the literature, various approaches have been studied for a broad variety of decoding problems. Among the most prominent are principal component analysis [86], linear discriminant analysis [87] and Bhattacharyya distance [88]. Besides these so-called filter methods, which are independent from a specific classifier (see Section 2.3), another type of routines—the so-called wrapper methods—works based on the idea of directly optimizing decoding accuracy by varying the included features. To do so, certain subsets of features are selected (e. g. using genetic algorithms or simple random selection) and the resulting decoding accuracy is evaluated with the desired classifier. This is repeated until the ideal set of features is determined (or after a pre-defined count of iterations).

Davies–Bouldin index A very prominent algorithm for feature selection is the so-called *Davies–Bouldin (DB) index* [89]. The DB index represents a cluster separation approach that measures the overlap of two clusters. Data points that shall be clustered may have arbitrarily high dimension S. For a set of N_κ S-dimensional feature vectors $\mathbf{v} \in \mathbb{R}^S$ the centroid $\boldsymbol{\mu}_\kappa$ of the corresponding cluster κ is computed as follows:

$$\boldsymbol{\mu}_\kappa = \frac{1}{N_\kappa} \sum_{\mathbf{v} \in \kappa} \mathbf{v}.$$

The cluster κ has the within-cluster scatter

$$\sigma_\kappa = \left(\frac{1}{N_\kappa} \sum_{\mathbf{v} \in \kappa} |\mathbf{v} - \boldsymbol{\mu}_\kappa|^2 \right)^{\frac{1}{2}}.$$

Based on the centroids and within-cluster scatters, a measure of the quality of clustering can be computed as follows:

$$R_{ij} = \frac{\sigma_i + \sigma_j}{\left\| \boldsymbol{\mu}_i - \boldsymbol{\mu}_j \right\|_2}, \quad i, j = 1, ..., K, \quad i \neq j.$$

For a total number of K clusters, the elements R_{ij} of the matrix $\mathbf{R} = (R_{ij}) \in \mathbb{R}^K$ describe the ratio between the within-class and between-classes scatter for class combination ij. Smaller values of R_{ij} indicate that the two clusters have less overlap in the feature space. This leads to two conclusions: First, for the same features, cluster pairs with low R_{ij} are easier to separate than those with higher values. Second, if different features (e. g. multiple channels) are available, R_{ij} can be computed for each individual feature; then the feature that leads to the smallest R_{ij} for a certain cluster pair yields most information to separate these two clusters. In the context of BCI, the number of clusters usually equals the amount of classes to be distinguished by the decoding routine.

2.3. Classification

An essential step in BCIs is to infer the intention of a user from their brain signal. As mentioned before, brain signals are usually acquired with a setup that contains multiple sensors (see Section 2.1). Each of the C sensors (typically called channels) samples data from a different spatial location with sampling rate f_{samp}. For a given measurement interval T, this results in a total of $S = T \cdot f_{\text{samp}}$ data samples. Let $\mathbf{a} = (a_{c,s}) \in \mathbb{R}^{C \times S}$ denote the acquired data. As described previously in Section 2.2.1, feature extraction (and selection) routines are applied to reduce the dimensionality of the data, which mostly means a reduction in both the number of channels $\tilde{C}$ and samples $\tilde{S}$ (i. e. $\tilde{C} < C$, $\tilde{S} < S$). The *classification* task then consists of assigning an estimated intention y_s of the user for each sampling point $s = 1, ..., \tilde{S}$ based on information that is contained in the resulting feature vector $\mathbf{x} = (x_{c,s}) \in \mathbb{R}^{\tilde{C} \times \tilde{S}}$.

More generally, a classifier (or decoder) Δ can be understood as a function that relates data features $\mathbf{x}$ (element of feature space $\mathfrak{F}$) to labels $\mathbf{y}$ (element of label space $\mathfrak{L}$):

$$\Delta : \mathfrak{F} \to \mathfrak{L}, \mathbf{x} \mapsto \mathbf{y}.$$

Depending on the desired application and the specific realization of the routines, $\mathfrak{F}$ and $\mathfrak{L}$ can differ accordingly (see Section 2.3.1). In general, all decoding routines work towards the same overall goals:

- high accuracy (i. e. high true positive rate, small false positive rate),
- high robustness (across different users, against poor signal-to-noise-ratio etc.),
- small delay/lag between intention and detection,
- fast computation times (ideally real-time capable).

The specific realization can be divided into different types of decoding, which will be explained in the following.

2.3.1. Types of decoding

Continuous vs. single trial

In the case described above, the decoding routine provides a prediction of the user's intention $\mathbf{y} = (y_s) \in \mathbb{R}^{\tilde{S}}$ for each time point. This scenario is referred to as *continuous decoding* (cf. Figure 2.1). Formally, a continuous decoder can be expressed as

$$\Delta^{\text{cont}} : \mathbb{R}^{\tilde{C} \times \tilde{S}} \to \mathbb{R}^{\tilde{S}}, \mathbf{x} \mapsto \mathbf{y}.$$

Here, it is assumed that the user intention—and hence, the assigned label—is a continuous parameter (e. g. degree of flexion of a certain finger or a target point in space). A broad variety of BCI scenarios uses discrete labels instead (e. g. class 1, class 2,... or thumb, index finger etc.). In this case, the label space changes to $\mathfrak{L} = \Omega^{\tilde{S}}$, where Ω denotes the finite set of label values (e. g. $\Omega = \{1, 2, 3\}$).

The majority of BCI approaches documented in the literature however, make use of a different strategy. Since brain signals typically have low signal-to-noise ratios, predicting the intention for every individual sample point constitutes a challenging decoding task. To circumvent this issue, the full sequence of measured data is epoched into smaller segments that typically contain a single event of interest (e. g. a single finger movement). These segments are called *trials*. Usually, trials are aligned with respect to a particular time point—like the presentation of a stimulus or the execution of a movement—and their length $S^{'} < \tilde{S}$ is chosen to contain all task-relevant brain responses. If the full data are epoched into N trials, the resulting dataset can be denoted as

$$\mathbf{X} = (\mathbf{x}_n) = (x_{c,s,n}) \in \mathbb{R}^{\tilde{C} \times S^{'} \times N}.$$

In this scenario, the decoding task reduces to the prediction of a single intention y_n for the entire trial n. For the full set of trials, predictions can be written as

$$\mathbf{Y} = (y_n) \in \Omega^N,$$

where Ω denotes the set of possible (discrete) labels. The routine—so-called *single trial decoding* (cf. Figure 2.1)—can be summarized as

$$\Delta^{\text{sngl.tr.}} : \mathbb{R}^{\tilde{C} \times S^{'} \times N} \to \Omega^N, \, \mathbf{X} \mapsto \mathbf{Y}.$$

Besides the requirement for a simplified decoding task, in many cases ground truth information is also unavailable on the level of individual samples. This can be due to the lack of corresponding measurements (e. g. movement tracking) or because the structure of the underlying problem favors an interval-based evaluation. Such constellations are addressed more conveniently with a single trial approach.

Single trial decoding is used primarily in *post-hoc* offline analysis. With appropriate adaptions, the approaches can also be used to perform (pseudo-)continuous decoding. To do so, a defined amount of incoming continuous data is cached and processed as if it were a single trial. This procedure is performed repetitively at intervals of fixed length ('sliding window'). Compared to an explicit continuous approach, single trial workarounds go along with some disadvantages. As classification is conducted on data segments of a certain length, decoding results are available only with a delay that is correlated to the specified window length (cf. Figure 2.1). The longer the chosen window, the higher the resulting delay between intention will become. On the other hand, long windows contain more data and thus, might provide a higher information content that can increase decoding precision. Consequently, a trade-off needs to be made between accuracy and immediacy of the predictions. In addition to delayed availability of decoding results, sliding window approaches are prone to introduce temporal blurring. Due to the fact that consecutive windows usually have considerable overlap in order to allow for a high rate of predictions, these windows share significant amount of mutual information. As a consequence, even short-lived events will likely be decoded in an extended range of consecutive windows. Especially in case of rapid events, this can become a severe limitation.

Static vs. dynamic Typically, brain signals as well as the features extracted from the acquired data are time sequences that contain temporal development of certain characteristics

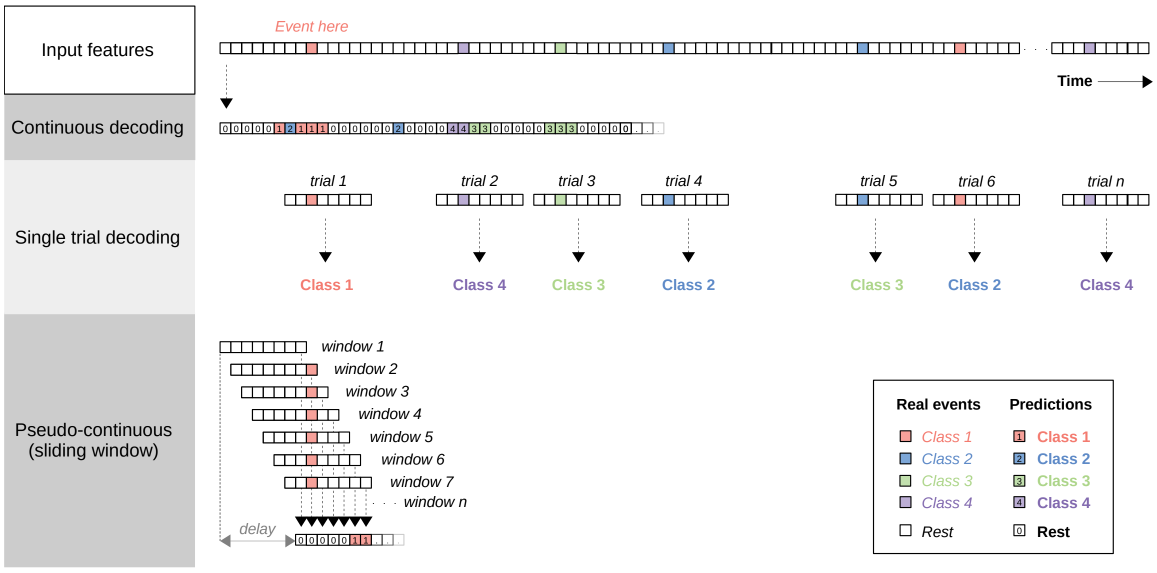


Figure 2.1: Scheme of differences between continuous and single trial decoding as well as the sliding window workaround for pseudo-continuous decoding with a single trial decoder.

like signal amplitude, spectral composition and so forth. Since data is further assessed from a series of different spatial locations, input data for classification is usually two-dimensional (feature/channel x time). The explanations below will be based on a single trial setting, i. e. $\mathfrak{F}^{\text{trial}} = \mathbb{R}^{\tilde{C}\times S'}$ for one individual trial and $\mathfrak{F} = \mathbb{R}^{\tilde{C}\times S' \times N}$ for all trials.

Static classifiers do not take into account the temporal component. Instead, all time points of a feature sequence are concatenated into a single feature vector, in which all entries are treated equally, i. e. $\mathfrak{F}^{\text{static}} = \mathbb{R}^{(\tilde{C}\cdot S')\times N}$. In doing so, the original time dimension is converted into a “normal” feature dimension (Figure 2.2). Hence, temporal development in the input data can no longer have immediate influence on the decoding result. The constructed single feature vector is classified to predict the class membership of the underlying data:

$$\Delta^{\text{static}} : \mathbb{R}^{(\tilde{C}\cdot S')\times N} \to \Omega^N, \, \tilde{\mathbf{X}} \mapsto \mathbf{Y}.$$

Prominent examples of static classifiers are linear discriminant analysis (LDA), most of the artificial neural network architectures (NN) and support vector machines (SVM). Over the last years, especially SVMs have been used extensively in brain signal decoding and became a gold-standard routine in the context of BCI (see Section 2.3.3).

Contrary to the static approaches, *dynamic classifiers* base their prediction on a sequence of feature vectors. Therefore, temporal relations within the sequence can be considered for classification [90, p. 28]. As it is likely that relevant neural processes are coded in the temporal structure of the data, this can play an important role for appropriate decoding of brain signals. Among the most prominent examples of dynamic classifiers is the hidden Markov model (HMM) decoder. In the field of automated speech recognition (ASR), HMMs constitute the gold-standard since the 1980s. Despite their large success in ASR, application

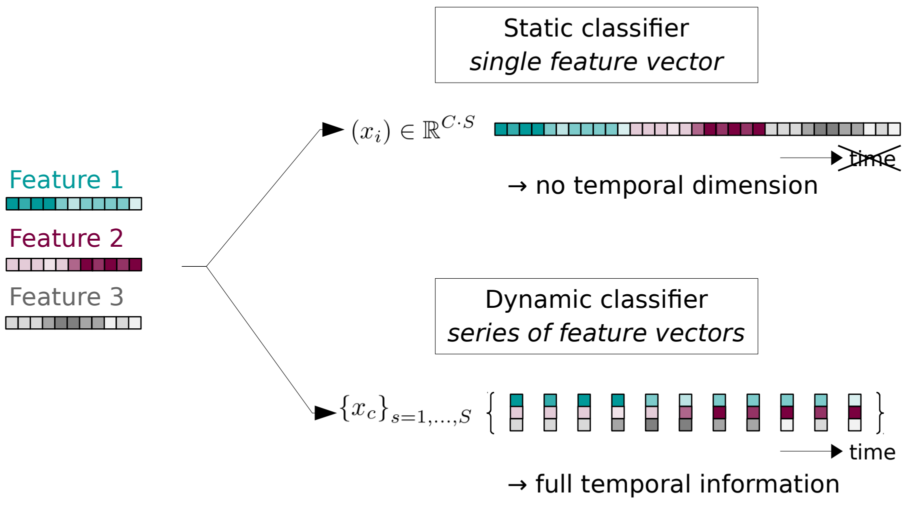


Figure 2.2: Scheme of feature processing in static and dynamic classifier approaches. The example depicts features from three channels ($\tilde{C} = 3$) and eleven samples ($S^{'} = 11$). The two-dimensional input data is reshaped into a one-dimensional vector in which all time samples are concatenated for static routines. Contrary, dynamic classifiers use the original time series as input.

of HMMs in the field of brain signal decoding is rather uncommon.

Synchronous vs. asynchronous Different from the previous comparisons, the distinction between *synchronous* and *asynchronous decoding* describes the application setting rather than technical aspects of the decoding routines itself. It is partially related to the continuous vs. single trial context.

In a *synchronous* setting, events always appear at pre-defined time points. Usually, this it realized by presentation of a stimulus to indicate when a subject shall perform the corresponding action. Alternatively, the stimulus can tell the user when to decide for a certain command that shall be executed. All data episodes in between stimuli are defined as rest or pause periods. An illustrative interpretation of this setting would be that the system is sensitive to commands only at well-defined time points. Consequently, decoding in a synchronous setting cannot contain false positive detections. In addition, since the expected time point of each event is known precisely, this information can be utilized in the signal processing chain. This significantly simplifies the decoding task and hence, typically results in higher decoding accuracies, simply as there is less room for errors. By concept, synchronous problems can be treated conveniently with single trial decoding approaches; this makes overall system design conceptually easier and in many cases, more robust. However, such a setting has the substantial disadvantage of not granting the user free control over the timing of the system, i. e. it is not entirely up to the user to decide when an action shall be initiated.

An *asynchronous* setting on the other hand allows for the generation of commands at arbitrary time points. To enable this, classification must be carried out continuously[3]. Since

[3]This can be done either by means of actual continuous decoding or using a single-trial workaround as

the possible positions of actual events are no longer known *a priori*, rest (or pause) episodes need to be identified correctly by the decoder in addition to determination of the type of a detected event. This circumstance introduces two additional sources of potential decoding errors: false positives and false negatives; with respect to ASR syntax, these will also be referred to as *insertions* and *deletions*, respectively. Usually, a trade-off between these error rates is inevitable. Most routines offer the option to control the balance between insertions and deletions by means of a parameter that determines the sensitivity of the decoder. Increasing decoder sensitivity lowers the false negative rate (i. e. it reduces the amount of deletions). Thus, it is likely that the majority of actual events will be detected, however, potentially at the cost of extensive amounts of insertions. Contrary, with too low sensitivity settings, the decoder might miss a significant number of commands. Depending on the intended application, different settings might be desired by the user. Besides the obvious advantage of free control over command timings for the user, asynchronous concepts can also operate without external stimulation, which may simplify the system setup in a variety of cases. As discussed, these benefits go along with substantially increased difficulty for robust decoding solutions.

2.3.2. Hidden Markov models

Hidden Markov models (HMM) are a specific type of stochastic signal model that describe a two-stage stochastic process. The theory behind the approach goes back to the work of Baum (e. g. [91]), Viterbi ([92]) and others. Since then it has been refined and summarized by a broad variety of books [93–95] and articles [33,34,95–99]. Potentially the most famous and influential work among them is the tutorial by Lawrence Rabiner from 1989 [100].

2.3.2.1. Theory

Definition As mentioned before, HMMs describe a two-stage stochastic process [95, p. 61]. First, the model represents a discrete stochastic process of transitions between so-called *states* S_i from a discrete, finite state space $S = \{S_1, S_2, ..., S_N\}$. The system changes states at regularly spaced, discrete time points $t = 1, 2, ...$ and the state of the system at time point t is denoted with q_t. Here, a first order Markov chain is considered, i. e. the process depends only on the current state and its immediate predecessor. That means the probability to find the system in state S_j at time point t is given by:

$$P\left(q_t = S_j | q_{t-1} = S_i, q_{t-2} = S_k, ...\right) = P\left(q_t = S_j | q_{t-1} = S_i\right).$$

Only stationary processes shall be considered, which means the probabilities are independent from the actual time point [100]. They can then be interpreted as a transition probability a_{ij} for the model to change states from i to j at any arbitrary[4] time point:

$$a_{ij} = P\left(q_t = S_j | q_{t-1} = S_i\right), \quad 1 \leq i, j \leq N.$$

described above (see 'Continuous vs. single trial').

[4]It is important to note that transition probabilities are time-independent. This means in particular that information on how long the model has already been in its current state cannot be considered.

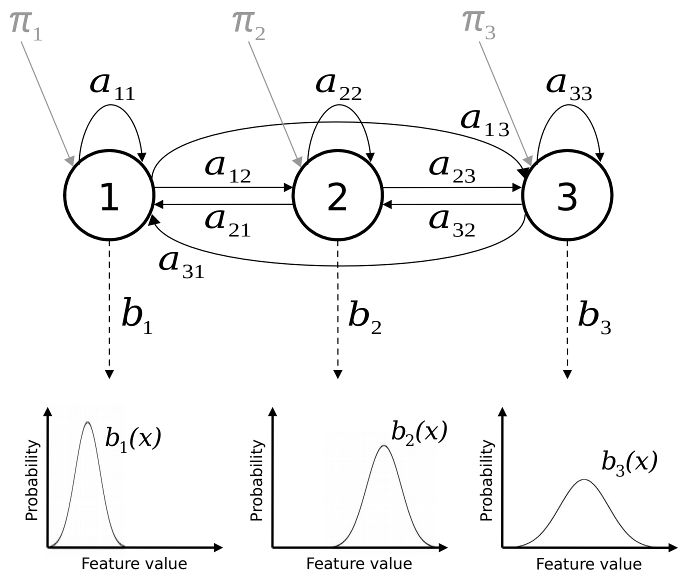


Figure 2.3: Structure of a fully-connected three-state hidden Markov model with emission probabilities b_j consisting of single Gaussians. Note that for means of simplicity, only one feature dimension is shown for $b_j(x)$; feature spaces are usually multi-dimensional.

Transition probabilities have the properties

$$a_{ij} \geq 0 \quad \text{and}$$
$$\sum_{j=1}^{N} a_{ij} = 1 \quad \forall i = 1, ..., N.$$

For the second stochastic component in the modeling process, an output (or observation) O_t is assigned to each time point with a certain probability. Outputs are strictly dependent only on the current state q_t, i. e.

$$P\left(O_t|O_{t-1}, O_{t-2}, ..., O_1\,,\, q_t, ..., q_1\right) = P\left(O_t|q_t\right).$$

This output sequence $\boldsymbol{O} = O_1, O_2, \ldots, O_T$ is the only thing that can be observed; hence, it is called observation sequence. The underlying state sequence $Q = q_1, q_2, \ldots, q_T$ is not accessible. In other words, the states are "hidden", which is the background of the terminology in "hidden Markov model".

In summary, the full hidden Markov model (first order) is defined by the following set of parameters:

- The set of states $S = \{S_1, S_2, ..., S_N\}$,
- transition probabilities a_{ij}, which can be represented by the transition matrix

$$\mathbf{A} = (a_{ij}) \in [0,1]^{N \times N}$$
$$\text{with } a_{ij} = P\left(q_t = S_j|q_{t-1} = S_i\right), \quad 1 \leq i, j \leq N,$$

- a so-called *prior* vector that describes the start probabilities

$$\begin{aligned}\boldsymbol{\pi} &= (\pi_i) \in [0,1]^N \\ \text{with } \pi_i &= P\left(q_1 = S_i\right), \quad i = 1, ..., N,\end{aligned}$$

- and the output probability distributions for each of the states

$$\begin{aligned}\boldsymbol{b}_i &= (b(o_k)) \in [0,1]^M, \quad \text{for each state } i = 1, ..., N. \\ \text{with } b_i(o_k) &= P\left(O_t = o_k | q_t = S_i\right), \quad k = 1, ..., M. \\ \Rightarrow \mathbf{B} &:= (b_{ik}) \in [0,1]^{N \times M} \\ \text{with } b_{ik} &= P\left(O_t = o_k | q_t = S_i\right).\end{aligned}$$

This definition assumes discrete output variables, i. e. $O_t \in \{o_1, o_2, ..., o_M\}$. Usually, these outputs o_k are interpreted as symbols. Hidden Markov models that are based on this type of output are called *discrete HMMs.* However, in the context of most signal analysis tasks, continuous output variables are required. To facilitate this, the emission probabilities $\boldsymbol{b}$ are modified. Given (n-dimensional) real-valued output vectors $\boldsymbol{x} \in \mathbb{R}^n$, the probabilities can be expressed as probability density functions

$$b_j(\boldsymbol{x}) = p\left(\boldsymbol{x} | q_t = S_j\right).$$

These types of models are called *continuous HMMs.* Such a formulation makes it necessary to find a way to describe the probability density functions. A common strategy is to approximate the density functions by a linear combination of normal distributions[5] [95, p. 64]

$$p\left(\boldsymbol{x}\right) = \sum_{m=1}^{\infty} c_m \mathcal{N}(\boldsymbol{x} | \boldsymbol{\mu}_m, \boldsymbol{\sigma}_m^2) \approx \sum_{m=1}^{M} c_m \mathcal{N}(\boldsymbol{x} | \boldsymbol{\mu}_m, \boldsymbol{\sigma}_m^2).$$

The approximation uses M so-called mixture components. For reasons of normalization of the probability density, the coefficients c_m must fulfill

$$\begin{aligned} c_m \in [0,1] \, \forall m \\ \sum_{m=1}^{M} c_m = 1.\end{aligned}$$

Using that definition, the emission probabilities of the HMM can be reformulated as

$$b_j(\boldsymbol{x}) = \sum_{m=1}^{M_j} c_{jm} \mathcal{N}(\boldsymbol{x} | \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm}^2).$$

In this most general case, every state contains an individual number of mixture components M_j. In addition to the parameters introduced above, this type of HMM also possesses

[5]The normal distribution is defined as: $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) := \left((2\pi)^n \det \boldsymbol{\Sigma}\right)^{-1/2} \exp\left(-\frac{1}{2}\left(\boldsymbol{x} - \boldsymbol{\mu}\right)^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \left(\boldsymbol{x} - \boldsymbol{\mu}\right)\right)$.

- for each state $j = 1, ..., N$ and each mixture $m = 1, ..., M$
  - a mean vector $\boldsymbol{\mu}_{jm} \in \mathbb{R}^n$,
  - the covariance matrix
$$\boldsymbol{\Sigma}_{jm} := \boldsymbol{\sigma}_{jm}^2 \in \mathbb{R}^{n \times n},$$
- as well as a (global) mixture matrix
$$\mathbf{C} := (c_{jm}) \in \mathbb{R}^{N \times M}.$$

Decoding The full (continuous) HMM is defined by $\lambda = \left\{\mathbf{A}, \boldsymbol{\pi}, \mathbf{C}, \left(\boldsymbol{\mu}_{jm}\right), (\boldsymbol{\Sigma}_{jm})\right\}$. Given that all parameters of a model λ have been estimated based on a training[6] set, the model can be used to compute the probability for the production of an observed sequence $\boldsymbol{O} = O_1, O_2, ..., O_T$, which will be denoted by $P(\lambda|\boldsymbol{O})$. If there is a set of K HMMs λ_k that have been trained for different classes, the production probabilities can be used as a decision criterion for classification of the data $\boldsymbol{O}$. Using Bayes' theorem, the *production probabilities* $P(\lambda_k|\boldsymbol{O})$ compute as follows:

$$P(\lambda_k|\boldsymbol{O}) = \frac{P(\boldsymbol{O}|\lambda_k)P(\lambda_k)}{P(\boldsymbol{O})}, \quad k = 1, ..., K.$$

To determine the class to which $\boldsymbol{O}$ most likely belongs, one searches the HMM that provides the maximum posterior probability for the observation of $\boldsymbol{O}$, i. e.

$$\lambda^* = \underset{\lambda_k}{\operatorname{argmax}}\ P(\lambda_k|\boldsymbol{O}) = \underset{\lambda_k}{\operatorname{argmax}}\ \frac{P(\boldsymbol{O}|\lambda_k)P(\lambda_k)}{P(\boldsymbol{O})} = \underset{\lambda_k}{\operatorname{argmax}}\ P(\boldsymbol{O}|\lambda_k)P(\lambda_k).$$

The last step uses that $P(\boldsymbol{O})$ is independent of λ_k and hence, does not have any influence on the maximization step. The probabilities $P(\lambda_k)$ are called prior probabilities, as they describe *a priori* knowledge on how likely certain models appear. If one assumes that these prior probabilities are equal for all classes (or can be neglected), the classification problem simplifies to

$$\lambda^* = \underset{\lambda_k}{\operatorname{argmax}}\ P(\boldsymbol{O}|\lambda_k). \tag{2.1}$$

This routine can be used to perform isolated unit decoding (or single trial decoding, cf. Section 2.3.1), i. e. classification of segmented data that each have a well-defined (single) class membership. In case of continuous decoding, the output probability of the entire data sequence alone hardly provides any useful information about individual events in the data. It is of great use to assess the state sequence $\boldsymbol{q} = q_1, q_2, ..., q_T$ that is associated with $P(\boldsymbol{O}|\lambda)$. Assuming this state sequence is known, one can infer conclusions from it about models that are involved and how the whole sequence segments into those different models. As the approach is a stochastic method, all evidence is of probabilistic nature only. Hence, there is not *the* state sequence that explains the data. Instead, one is interested in specific

[6] Details on the training algorithms are omitted here as they are not essential for the understanding in this context. Comprehensive presentation of training procedures can be found in [100] or [95, pp. 76-90].

sequences. A natural choice is to look for the most likely state sequence, i. e. the sequence $\boldsymbol{q}^*$that—for a given model λ and an observation $\boldsymbol{O}$—provides the highest likelihood P:

$$\boldsymbol{q}^* = \underset{\boldsymbol{q}}{\text{argmax}}\ P(\boldsymbol{O}, \boldsymbol{q}|\lambda). \tag{2.2}$$

This state sequence is known as the *Viterbi path* (see below).

Of central relevance for all findings presented so far is the computation of the production probabilities $P(\boldsymbol{O}|\lambda)$. Assuming an observation sequence of length T, i. e. $\boldsymbol{O} = O_1, O_2, ..., O_T$, the model goes through a corresponding state sequence $\boldsymbol{q} = q_1, q_2, ..., q_T$ to generate these observations. Consequently, the production probability is computed as the product of all involved emission probabilities b_{q_t} along this sequence $\boldsymbol{q}$:

$$P(\boldsymbol{O}|\boldsymbol{q}, \lambda) = \prod_{t=1}^{T} b_{q_t}(O_t). \tag{2.3}$$

For Eq. (2.3) it is assumed that the model goes through the specific state sequence $\boldsymbol{q}$. Hence, to compute the probability for $P(\boldsymbol{O}, \boldsymbol{q}|\lambda)$, one must consider the probability for the model to use that particular sequence in the first place. This is computed as the product of the prior probability π_{q_1} to start in state q_1 and all transition probabilities a_{ij} required to generate the rest of the sequence:

$$P(\boldsymbol{q}|\lambda) = \pi_{q_1} \prod_{t=2}^{T} a_{q_{t-1}q_t} \overset{(a_{0t}:=\pi_t)}{=} \prod_{t=1}^{T} a_{q_{t-1}q_t}. \tag{2.4}$$

As a result, $P(\boldsymbol{O}, \boldsymbol{q}|\lambda)$ can be computed as follows:

$$P(\boldsymbol{O}, \boldsymbol{q}|\lambda) = P(\boldsymbol{q}|\lambda) \cdot P(\boldsymbol{O}|\boldsymbol{q}, \lambda) \overset{(2.3),(2.4)}{=} \prod_{t=1}^{T} a_{q_{t-1}q_t} b_{q_t}(O_t).$$

By design, a (continuous) HMM can reproduce a given observation sequence $\boldsymbol{O}$ from any arbitrary state sequence $\boldsymbol{q}$.[7] Consequently, to compute the total probability of producing the observation $\boldsymbol{O}$ with a certain model λ, all possible state sequences must be taken into account. The joint probability is a sum of all individual paths:

$$P(\boldsymbol{O}|\lambda) = \sum_{\boldsymbol{q}} P(\boldsymbol{O}, \boldsymbol{q}|\lambda) = \sum_{\boldsymbol{q}} P(\boldsymbol{q}|\lambda) P(\boldsymbol{O}|\boldsymbol{q}, \lambda). \tag{2.5}$$

Calculation of $P(\boldsymbol{O}|\lambda)$ by means of Eq. (2.5) is computationally highly expensive ($\mathcal{O}(TN^T)$, [95, p. 69]) as it requires considering all possible state sequences. Fortunately, computation of $P(\boldsymbol{O}|\lambda)$ can be simplified significantly by an efficient algorithm that makes use of the Markov property, the so-called *forward algorithm*.[8] For that, one defines forward variables

[7]This is due to the fact that the assumed normal distributions do not vanish at any point, i. e. $\nexists \boldsymbol{x} : b_j(\boldsymbol{x}) = 0$. The resulting probabilities can of course become extremely small.

[8]A detailed description can be found in [95].

$\alpha_t(S_i)$ describing the probability that the model λ produced the observation sequence $O_1, O_2, ..., O_t$ and is exactly in state S_i at time point t.

$$\alpha_t(S_i) := P(O_1, O_2, ..., O_t,\, q_t = S_i|\lambda)$$

A recursive algorithm can be formulated to compute all $\alpha_t(S_i)$:

$$\begin{aligned}\alpha_1(S_i) &= \pi_{S_i} b_{S_i}(O_1) \\ \alpha_{t+1}(S_j) &= \sum_{i=1}^{N} \alpha_t(S_i)\, a_{ij} \cdot b_j(O_{t+1})\,. \end{aligned} \tag{2.6}$$

Using Eq. (2.6), the final production probability can be computed as the sum of the forward probabilities for all states at the end of the sequence $t = T$:

$$P(\boldsymbol{O}|\lambda) = \sum_{i=1}^{N} \alpha_T(S_i)\,. \tag{2.7}$$

Compared to the brute-force solution (Eq. (2.5)), computation of $P(\boldsymbol{O}|\lambda)$ by means of the forward algorithm drastically reduces the complexity of the problem. Instead of $\mathcal{O}\left(TN^T\right)$, the computational effort of Eq. (2.7) is only $\mathcal{O}\left(N^2T\right)$ [100].

Another strategy to simplify the computation in Eq. (2.5) is constraining the sum from running over all possible state sequences to a smaller subset. The rationale behind this is that the majority of sequences—although theoretically able to produce the observation—have such a small probability that their contribution to the sum is negligible. In the extreme case, only the sequence with the highest contribution is regarded. As mentioned above, this sequence is called Viterbi path $\boldsymbol{q}^*$ (cf. Eq. (2.2)). It can be computed using an approach that is conceptually similar to the forward variables. To do so, one first defines quantities

$$\delta_t(S_i) := \max_{q_1, q_2, \cdots, q_{t-1}} P(q_1, q_2, ..., q_{t-1}, q_t = S_i,\, O_1, O_2, ..., O_t|\lambda)$$

that describe the highest probability for a certain path, which generates the observations up to time point t and ends in state S_i. In this setting, the HMM λ, the entire observation sequence $O_1, O_2, ..., O_t$ as well as the final state q_t are fixed. Maximization is performed with respect to the initial $t-1$ states $q_1, q_2, ..., q_{t-1}$. Consecutive values can be computed by induction [100]

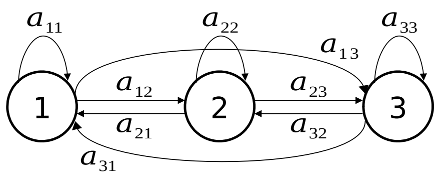
$$\delta_{t+1}(S_j) = \max_{S_i}\left(\delta_t(S_i) \cdot a_{ij}\right) \cdot b_j(O_{t+1})\,.$$

To assess the entire state sequence $\boldsymbol{q}^*$ that leads to the final optimal probability $\delta_T(S_i)$, the states need to be tracked for each recursion step , i. e.

$$\psi_t(S_j) = \operatorname*{argmax}_{S_i} \delta_{t-1}(S_i)\, a_{ij}, \quad t = 2, ..., T,\ j = 1, ..., N.$$

The final state can be determined as the maximum among $\delta_T(S_i)$ and with the help of ψ_t, the rest of the state sequence can be backtracked:

$$\begin{aligned}\boldsymbol{q}_T^* &= \operatorname*{argmax}_{S_i} \delta_T(S_i) \\ \boldsymbol{q}_t^* &= \psi_{t+1}\left(\boldsymbol{q}_{t+1}^*\right), \quad t = T-1, T-2, ..., 1.\end{aligned}$$



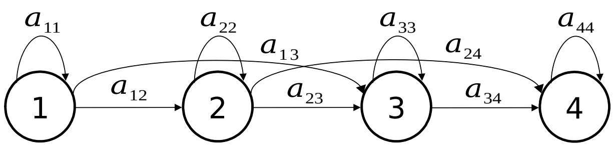


Figure 2.4: Scheme of two different HMM topologies: (a) fully-connected (ergodic) three-state model, (b) four-state HMM with Bakis topology.

Coming back to the initial idea of simplifying the computation of the overall production probability (Eq. (2.5)), one can now use the Viterbi path $\boldsymbol{q}^*$ to compute an estimate

$$P(\boldsymbol{O}|\lambda) = \sum_{\boldsymbol{q}} P(\boldsymbol{O}, \boldsymbol{q}|\lambda) \approx P(\boldsymbol{O}, \boldsymbol{q}^*|\lambda) = \max_{S_i} \delta_T(S_i) \, .$$

2.3.2.2. Specific requirements

As presented in the previous section, an HMM is defined by a set of parameters $\lambda = \left\{\mathbf{A}, \boldsymbol{\pi}, \mathbf{C}, \left(\boldsymbol{\mu}_{jm}\right), (\boldsymbol{\Sigma}_{jm})\right\}$. For a topology with N states, M mixture components and n-dimensional features, the parameter set has

$$\underbrace{N^2}_{\mathbf{A}} + \underbrace{N}_{\boldsymbol{\pi}} + \underbrace{NM}_{\mathbf{C}} + \underbrace{NM \cdot n}_{\boldsymbol{\mu}} + \underbrace{NM \cdot n^2}_{\boldsymbol{\Sigma}} = N \cdot \left[N + 1 + M \cdot \underbrace{(1 + n + n^2)}_{\text{features}} \right] \tag{2.8}$$

values. This term is quadratic in both number of states and feature dimension. Hence, it can easily accumulate to a tremendously high number[9] of parameters. As these must all be determined based on training data—which are rather limited in most BCI studies—reduction of the parameter count is of high importance to ensure meaningful estimates and to avoid overfitting. Lowering the total number parameters can be done in several ways, which can more or less be separated in *model constraints* and *feature selection* approaches.

Model constraints aim at tailoring the properties and topology of the model to the desired purpose. An unconstrained model allows for state transitions between all states. This is called an *ergodic*[10] model (cf. Figure 2.4 (a)). For situations in which a certain (causal) chronological structure of the signal is assumed, it might be more reasonable to use a so-called *left-to-right topology*. In that case, the transition matrix $\mathbf{A}$ is restricted to an upper

[9]In the exemplary case of $N = 5$, $M = 3$ and $n = 64$ (typical number of electrodes in ECoG recordings), Eq. (2.8) yields a total of 62445 parameters.

[10]The term ergodic does not necessarily require connections between all states. It is sufficient if the topology allows to reach any state of the model from any other state within a finite number of steps. Full transition matrices are a special case for which this is fulfilled already with a single step.

triangular shape, i. e.

$$\mathbf{A} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,N} \\ 0 & a_{2,2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{N-1,N} \\ 0 & \cdots & 0 & a_{N,N} \end{pmatrix},$$

which implies that state changes always lead to higher state numbers. Additionally, forward jumps are often restricted to a certain allowed range of Δ steps:

$$a_{ij} = 0 \, \forall j : \, j > i + \Delta.$$

A frequently used choice is $\Delta = 2$, which is referred to as *Bakis topology* (cf. Figure 2.4 (b)). Constraining state transitions reduces the number of non-zero elements in the transition matrix, and by that decreases the model parameter count. The impact of constrained transition topologies goes beyond mere reduction in the count of parameters though. As it introduces restrictions on state changes, it also eliminates a significant number of state sequences an ergodic model would be able to go through [95, p. 128].

Besides on state transitions, it is also possible to impose restrictions on the covariances $\boldsymbol{\Sigma}_{jm}$. A common strategy is to use a diagonal shape instead of full matrices [95, p. 140]. This implies that individual features are uncorrelated. Due to the quadratic growth of the number of elements in $\boldsymbol{\Sigma}_{jm}$ with the feature dimension in full matrices, these account for a large share of the total number of model parameters. Restrictions substantially limit this contribution; instead of n^2 parameters for each state (and mixture) in full covariance matrices, only n values need to be estimated in case of diagonally shaped $\boldsymbol{\Sigma}_{jm}$. However, applicability of this constraint depends on how well the actual data fulfill the assumption of being uncorrelated. Optionally, a variety of approaches (e. g. principal component analysis) can be used to de-correlate input features if necessary.

As seen in Eq. (2.8), the number of features plays a crucial role ($\mathcal{O}\left(n^2\right)$) for the total amount of parameters of HMM decoders. Most acquisition modalities record brain signals from a rather high number of channels. Usually, only a subset of those channels provides meaningful information for the decoding task, which is mainly due to the fact that the neural ensembles involved in a particular task typically cover narrow brain areas. Hence, selecting the relevant channels not only reduces the number of parameters substantially, but also removes irrelevant information that could otherwise bias the model erroneously. Some feature selection approaches are discussed in Section 2.2.2 and all modifications used in this work are explained in Section 2.2.2.

After a suitable model topology has been specified, model parameters need to be initialized before parameter refinement can be performed based on training data. Proper initial values are fundamental to avoid poor local minima during optimization. Initialization strategies used in this work will be presented in Section 3.3.

2.3.2.3. Benefits

HMMs are used to perform probabilistic dynamic classification. Hence, decoding results are available in the form of true probabilities, representing how well the observation fits to the assumed model. This does not only facilitate easier result interpretation and allow for discarding of uncertain predictions (e. g. by means of thresholding) to decrease false positive detection rates, but also does it provide the opportunity to incorporate additional information into the decoding process. Usually, prior knowledge on certain characteristics of a system is available in the form of probabilities for the possible output realizations. These can be considered conveniently in an HMM-based decoder by using them as an adjustment for the computed production probabilities.

By design, HMMs are intended to model the temporal structure of signals. Due to the possibility of state loops, models are able to account for variations in the temporal extension of the underlying signal patterns.[11] Such differences can hardly by captured by static approaches and thus, easily lead to wrong decoding results. Compared to simple class assignment approaches, direct signal modeling can also provide insight into signal processes that might help to develop theoretical descriptions of underlying mechanisms; and it might provide a means to understand properties of signal sources without having it available [100].

The combination of all above mentioned benefits makes HMMs the gold-standard choice when it comes to automated speech recognition [33, 34, 93, 99, 101] as well as a frequently used method for handwriting recognition [102], bioinformatics (e. g. analysis of biological sequences, like DNA) [103, 104] and various other topics of temporal pattern recognition.

2.3.2.4. Difficulties in the context of brain signal decoding

It has been mentioned before that HMMs are a thoroughly-studied and well-established technique in the field of speech decoding. From feature extraction techniques, over specific model topologies, up to sophisticated language models, almost every challenge could be addressed appropriately when it comes to the application of HMMs in ASR. However, much less profound results are available as far as brain signal decoding with HMMs is concerned. Compared to the analysis of speech signals, BCI applications differ in numerous aspects and consequently, pose a series of difficulties, a selection of which is listed in the following:

Multiple sensors Speech recognition systems use input from a single audio channel (microphone). Contrary, in a BCI scenario, data are typically acquired from several sensors (e. g. electrodes).

Specific features Typical ASR features (e. g. cepstral coefficients) are not suited for brain signals.

Fundamentals The processes involved in speech production are well-known (anatomy of vocal tract, acoustics, typical sound of phonemes etc.). The underlying mechanisms in the brain, on the other hand, are not entirely well-understood. So far, there is no profound knowledge on sub-structure of brain signals (like the analogy to phonemes, syllables and words).

[11]In the case of speech decoding this can e. g. refer to stretching or clipping of certain syllables.

Training data For most BCI studies, there are substantial difficulties in acquiring large training data bases (esp. for invasive data).

Cross-subject The comparability of data across subjects is much more complicated than for speech signals. Many people speak the same language, but individual brains can differ substantially with respect to differences in localization of involved brain areas as well as the mechanisms of information processing themselves.[12]

Prior knowledge The construction of generalized prior knowledge is difficult compared to the use of dictionaries or grammar models.

2.3.3. Support vector machines

The *support vector machine* (SVM) is a discriminative, static classifier that—without extensions—performs binary classification. By default, SVMs make no assumptions on the distribution of the data. As typical for static classifiers (cf. Section 2.3.1), all feature values are considered as a single high-dimensional input vector $\mathbf{x} \in \mathbb{R}^F$. In particular, this has the consequence that feature values from all time points end up in the same feature space that does not distinguish between "time" and "feature" dimension. Each observation (e. g. each trial) is described by its feature vector and represented in the feature space as one single point. In the simplest case, observations are labeled to belong to one of two classes with label values $y_n \in \Omega = \{-1, 1\}$. For the purpose of classification, the goal is to compute the hyperplane that optimally separates the points of one class from those belonging to the other class. The hyperplane is given by

$$\mathbf{w} \cdot \mathbf{x} + b = 0,$$

where $\mathbf{w}$ is the weight vector that is oriented normal to the hyperplane and b is an offset term [106]. The perpendicular distance between the hyperplane and the origin is $b/\left\|\mathbf{w}\right\|$. Let d_+ and d_- denote the (perpendicular) distance from the hyperplane to the closest positive and negative data point, respectively. The sum of these two distances $m = d_+ + d_-$ is called *margin*. With respect to finding "the hyperplane that optimally separates the points", the term "optimally" means maximization of this margin m. To formulate the optimization task, the following constraints are introduced:

$$\begin{aligned} \mathbf{w} \cdot \mathbf{x}_n + b &\leq -1 \text{ for } y_n = -1 \\ \mathbf{w} \cdot \mathbf{x}_n + b &\geq +1 \text{ for } y_n = +1. \end{aligned}$$

These constraints suggest that the data is linearly separable, i. e. a hyperplane exists such that all data points $\mathbf{x}_n$ with label $y_n = -1$ strictly lie on one side of the plane, whereas all points belonging to the other class are on the opposite side. The above mentioned constraints can be combined to:

$$y_n \left(\mathbf{w} \cdot \mathbf{x}_n + b\right) - 1 \geq 0, \ \forall n. \tag{2.9}$$

[12]For specific types of BCI applications (e. g. motor imagery based), it is even known that some people are fully unable to control these systems. This is referred to as BCI illiteracy [105].

Without loss of generality[13], it is assumed that data points (at least one) exist for which the first/second constraint holds exactly, i. e. the points are on the hyperplane

$$H_{1/2} : \mathbf{w} \cdot \mathbf{x}_n + b = \mp 1.$$

In this case, the distance between the hyperplane(s) and the origin is $|\mp 1 - b| \,/\, \|\mathbf{w}\|$. The resulting margin is

$$m = d_+ + d_- = 1/\, \|\mathbf{w}\| + 1/\, \|\mathbf{w}\| = 2/\, \|\mathbf{w}\| \,. \tag{2.10}$$

Apparently, both hyperplanes are parallel since they have the same normal $\mathbf{w}$. As enforced by constraint (2.9), no data points lie within the margin area. Consequently, minimizing the norm of the normal $\|\mathbf{w}\|$ with respect to constraint (2.9) yields the pair of optimal hyperplanes H_1 and H_2 (as it maximizes the margin m, cf. Eq. (2.10)). The data points that fulfill the equality in Eq. (2.9)—and thus, lie exactly on the hyperplanes $H_{1/2}$—are called *support vectors*. To determine the class membership of an unknown data point $\mathbf{x}^*$, one considers the plane halfway between H_1 and H_2

$$H_s : \mathbf{w} \cdot \mathbf{x} + b = 0$$

and computes on which side of H_s the test sample $\mathbf{x}^*$ lies. That means, the assigned label will be $y^* = \operatorname{sign}(\mathbf{w} \cdot \mathbf{x}^* + b)$.

As mentioned above, such separation can only be successful if data is linearly separable; otherwise, constraint (2.9) is violated. To facilitate the use in situations in which the demand for linear separability is not fulfilled, the routine can be extended by introduction of so-called *slack variables* ξ_n. The constraints (2.9) are reformulated with respect to the introduced slack variables:

$$\begin{aligned} y_n (\mathbf{w} \cdot \mathbf{x}_n + b) &\geq (1 - \xi_n) \\ \xi_n &\geq 0, \ \ \forall n. \end{aligned} \tag{2.11}$$

In this case, a data point can lie on the wrong side of the separation plane—which means it is a decoding error—if its slack variable is $\xi_n > 1$. That means that the sum over all slacks $\sum_n \xi_n$ defines an upper bound for the number of such error points. It can be introduced to the cost function (for optimization of the hyperplane H_s^C) as an additional term that penalizes classification mistakes. The resulting optimization task can be expressed as [107]:

$$\begin{aligned} &\min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2} + C \cdot \sum_n \xi_n(\mathbf{w}, b) \\ \text{w.r.t. } &\xi_n \geq 0. \end{aligned}$$

The parameter C is used to control the influence of the slack term in order to adjust the trade-off between the highest possible margin and the lowest possible number of misclassifications. In doing so, the generalizability of the routine can be controlled (Figure 2.5). Another extension of the approach allows for the use of non-linear separation bounds.

[13]The assumption can always be fulfilled by choosing an appropriate scale for $\mathbf{w}$ and b [106].

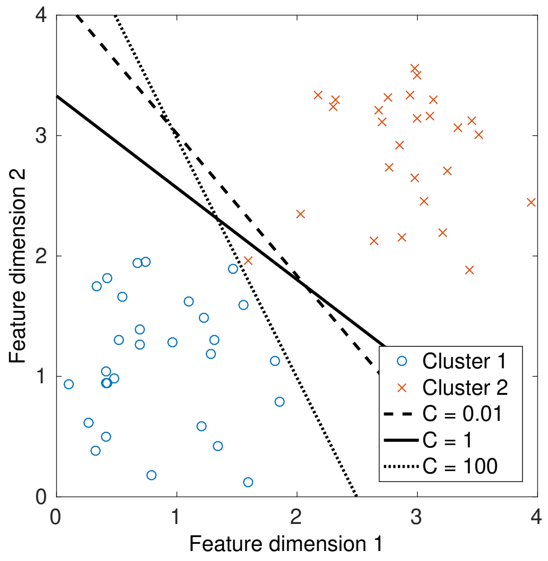


Figure 2.5: Different SVM training results with varying slack parameter C. The plot shows the separating hyperplane H_s^C with three different choices for C, namely $C = \{0.01, 1, 100\}$. Data of two clusters are simulated each with 30 individual (x, y)-tuples drawn from a normal distribution $\mathcal{N}(\mu_{1/2}, \sigma = 0.5)$; with $\mu_1 = 1$, $\mu_2 = 3$. Apparently, the resulting clusters are linearly separable. Still, the three hyperplanes provide insight into the different behavior of the routine when varying the slack term. The case $C = 0.01$ hardly considers the errors at all. This is reflected by the resulting plane $H_s^{0.01}$ (dashed line) that involves one error ('x' point at $(1.59, 1.96)$) and has a strong tendency to maximize the margin. Contrary, for $C = 100$ a very strong focus lays on not having erroneous data points. Due to the linear separability in this example, the plane H_s^{100} (dotted line) separates both clusters without any mistakes. However, this result is highly susceptible to outliers from Cluster 1 that may appear eventually in test data. Hence, less training mistakes are achieved with the consequence of loss of generalization. As expected, H_s^1 (solid line) represents a compromise between both extremes.

This is done using the so-called *kernel trick*. Instead of separating features directly in the original feature space, feature vectors are transformed into a higher dimensional space using a kernel function [108]. Details on this approach are omitted here, as only linear SVMs are used in this work. Comprehensive overviews on the method can be found e. g. in references [106, 107, 109].

As mentioned earlier, the straightforward SVM approach provides binary classification. For multi-class problems, two different strategies are commonly used: the *one-vs–one* and *one-vs-all* mode. In one-vs-one mode, individual binary classifications are carried out for each of the possible "class duels". This sums up to a total amount of $K \cdot (K - 1)/2$ classification procedures to decide a K-class problem. The class with the highest number of "wins" in these individual one-vs-one duels is selected as overall winner of the K-class problem. The one-vs-all mode requires only K classification procedures to solve the K-class problem. In this approach, data of a particular class are classified against data from all remaining classes. This is done once for each of the classes. Finally, the class that ends up with the highest output function value decides the classification problem in its favor. Contrary to the one-vs-one mode, calibration of the output functions is required to achieve meaningful results in a one-vs-all setting [106].

2.3.4. Prior-knowledge incorporation

In most cases, lots of information is available on actions that are performed frequently (or preferentially) by the user. This *prior knowledge* (PK) might be used to improve decoding accuracies in BCI applications and thereby increase the robustness of such systems. Besides mere improvements in accuracy, another important goal of PK incorporation is to preserve full flexibility of possible decoding outcome. To be more specific, this means that frequently appearing events—or events that the user wishes to appear more frequently than others—shall be emphasized, but decoding output shall not be limited to a set of pre-defined command sequences.[14]

Different strategies can be used to incorporate PK into the decoding, depending mainly on the specific type of available PK. In this work, PK incorporation is investigated in a setting that provides information on the frequency of certain event sequences.[15] The usual approach to consider this type of information is *n-gram modeling.* An n-gram is a sequence of n consecutive elements (e. g. phonemes, words). The n-gram model is a probabilistic model that uses the information on typical frequencies of appearance of the individual n-grams to make a prediction of the next event. More specifically, based on the history of the previous $n-1$ elements, the likelihood of the n'th element is approximated. For $n = 1$, this refers to the simple case of just considering how frequent individual elements are to appear. Especially in speech decoding, the use of *bi-grams* ($n = 2$) and *tri-grams* ($n = 3$) is very common, whereas higher order models are hardly used [95, p. 96].

To incorporate the information from an n-gram model into the decoding, one considers the following setting: Assuming that the user wants to "perform" a certain sequence of elements (or a chain of basic commands) $\boldsymbol{\omega} = \omega_1, \omega_2, ..., \omega_N$, the corresponding likelihood of that particular sequence is encoded in the n-gram model as $P^{\text{n-gram}} := P(\boldsymbol{\omega})$. To translate this intention into practice, the sequence $\boldsymbol{\omega}$ is transformed into a signal $\mathbf{a}$. This would, for example, correspond to brain activity in case of a BCI setting or to an acoustic signal (i. e. speech) in speech decoding (Figure 2.6). This signal is then recorded and features $\mathbf{x}$ are extracted from it. The likelihood of observing $\mathbf{x}$, when the underlying sequence was $\boldsymbol{\omega}$, is given by $P(\mathbf{x}|\boldsymbol{\omega})$. This likelihood can be modeled, e. g. by means of HMMs ($P^{\text{HMM}} := P(\mathbf{x}|\boldsymbol{\omega})$). For decoding, one is interested in the event sequence $\boldsymbol{\omega}^*$ that is most likely to explain the observed features, i. e.

$$\boldsymbol{\omega}^* = \underset{\boldsymbol{\omega}}{\operatorname{argmax}}\, P(\boldsymbol{\omega}|\mathbf{x}). \tag{2.12}$$

Note that here—although it looks conceptually similar to the Viterbi path for HMMs (cf. Section 2.3.2)—the sequence $\boldsymbol{\omega}^*$ does not refer to internal states within an HMM, but to the series of events in the data.

[14]In speech decoding, the analogue thereof is to assign higher weights to phoneme constellations that form meaningful words, while retaining the ability to spell individual letters or digits (e. g. for phone numbers).

[15]This is very similar to the situation in speech decoding, in which *a priori* knowledge on the frequency of appearance of phoneme combinations is available.

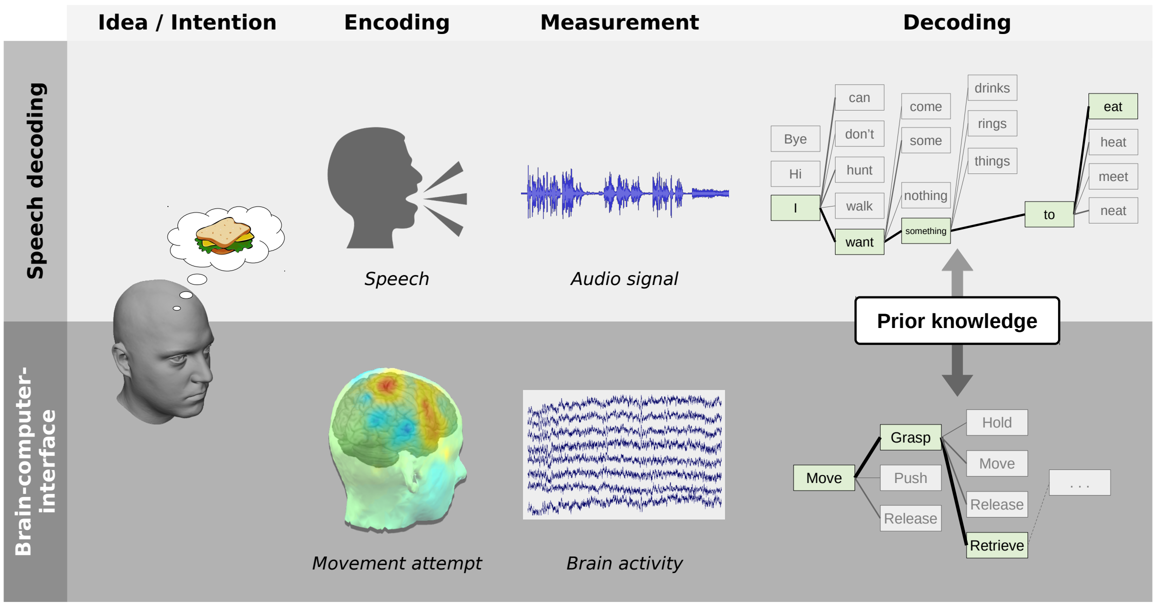


Figure 2.6: Scheme of a typical decoding setting with prior knowledge incorporation in speech decoding and brain-computer-interfaces. (Sub-figure "Speech" reproduced from emojione project (https://github.com/emojione/emojione) [CC BY-SA 4.0], via Wikimedia Commons.)

Using Bayes' theorem, Eq. (2.12) can be rewritten as

$$\boldsymbol{\omega}^* = \underset{\boldsymbol{\omega}}{\operatorname{argmax}} \frac{P(\boldsymbol{\omega})P(\mathbf{x}|\boldsymbol{\omega})}{P(\mathbf{x})}. \tag{2.13}$$

This maximization does not depend on $P(\mathbf{x})$, hence, Eq. (2.13) can be simplified to

$$\boldsymbol{\omega}^* = \underset{\boldsymbol{\omega}}{\operatorname{argmax}} P(\boldsymbol{\omega})P(\mathbf{x}|\boldsymbol{\omega}) = \underset{\boldsymbol{\omega}}{\operatorname{argmax}} P^{\text{total}}(\mathbf{x}, \boldsymbol{\omega}).$$

The total probability P^{total} required for computation of the likeliest sequence is a simple product of the n-gram likelihood and the observation (or production) probability of the HMMs

$$P^{\text{total}}(\mathbf{x}, \boldsymbol{\omega}) := P(\boldsymbol{\omega}) \cdot P(\mathbf{x}|\boldsymbol{\omega}) = P^{\text{n-gram}} \cdot P^{\text{HMM}}.$$

In practice, however, it is typical that the dynamic range of both factors differs significantly [95]. To avoid that the decoding is solely dominated by one of the components (n-gram or HMM), a weighting constant ρ is introduced [94]:

$$P^{\text{total}} = (P^{\text{n-gram}})^{\rho} \cdot P^{\text{HMM}}.$$

In speech decoding, this constant is referred to as linguistic matching factor and needs to be determined empirically for each specific context [95].

3. Material and Methods

In this chapter, all routines that are used repeatedly throughout the investigations in this work are presented (Sections 3.1–3.6). Additionally, experimental paradigms and details on data recording are introduced in Section 3.7. Detailed information on the data and all methods that are specific to individual parts of the results are given in the corresponding result sections.

3.1. Feature extraction routines

A broad variety of features is documented in the literature (cf. Section 2.2.1). Since the focus of this work is on signal classification, two commonly used types of features are utilized. These features have been selected to cover two different characteristics of the signal, namely time and frequency domain features (or temporal and spectral features, respectively). *Time domain features* are investigated as they immediately represent temporal variation of signal amplitudes and are widely used in the analysis of neural phenomena (e. g. event related potentials (ERP)). The main information of time domain features originates from the signal phase. *Frequency domain features* track power fluctuations in specific frequency bands. Hence, they represent a signal property that is independent from explicit phase information. As mentioned before (Section 2.2.1), a broad variety of studies demonstrated a particularly substantial information value of the so-called *high gamma band*—this refers to signal components from the frequency interval of approximately 60–200 Hz—for various decoding scenarios.

Both features are computed using simple and straightforward algorithms, which will be presented in the following.

3.1.1. Low-frequency time domain features

Low-frequency time domain features (LFTD) represent a low-pass filtered version of the original data. Filtering is performed by means of a simple approach based on fast Fourier transform (FFT). Raw data $\mathbf{a} = (a_{c,s}) \in \mathbb{R}^{C\times S}$ are transformed into the Fourier-domain and all coefficients $\hat{a}_c(\omega_k)$ above a pre-defined threshold ($\omega_k > \omega_{\text{cut-off}}$) are set to zero:

$$\hat{a}_c(\omega_k) = \mathcal{F}\left[a_{c,s}\right](\omega_k) \cdot \theta(\omega_{\text{cut-off}} - \omega_k), \quad c = 1, ..., C.$$

Here, $\mathcal{F}$ denotes the (discrete) Fourier transform and θ the Heaviside step function. The inverse Fourier transform is applied to transfer back the data to the time domain:

$$\tilde{x}_{c,s} = \mathcal{F}^{-1}\left[\hat{a}_c(\omega_k)\right], \quad c = 1, ..., C.$$

Subsequently, the resulting sequence is downsampled by keeping every d_s-th sample (downsampling factor $d_s \in \mathbb{N}$) and discarding the rest to compute the final feature values

$$\mathbf{x}^{\mathrm{LFTD}} = \left(x^{\mathrm{LFTD}}_{c,n}\right) = \left\{\tilde{x}_{c,1+(n-1)\cdot d_s}\right\}_{c=1,\ldots,C,\, n=1,\ldots,\left\lceil \frac{S}{d_s}\right\rceil}.$$

In summary, LFTD feature calculation depends on two parameters: cut-off frequency $\omega_{\text{cut-off}}$ and downsampling factor d_s.

3.1.2. High gamma features

Frequency domain features are extracted using a sliding window approach. Data $\mathbf{a} = (a_{c,s}) \in \mathbb{R}^{C\times S}$ are divided into segments with a defined length (window size S_w). A shift parameter Δ_w is introduced to specify by how many samples data are shifted between adjacent windows. This results in a total of $N_w = \lfloor (S - S_w)/\Delta_w \rfloor + 1$ windows.

All individual segments are multiplied with a window function $\boldsymbol{w} = (w_s) \in \mathbb{R}^{S_w}$ (Hann window) to reduce sidelobes in the computed spectrum. The resulting signal of the n-th window is given by

$$\tilde{a}^n_{c,s} = a^n_{c,s} \cdot w_s, \quad c = 1, \ldots, C,\; s \in S^n,$$

whereby S^n denotes the set of data samples that belong to the n-th window.

The periodogram of the signal is estimated for each window—computed individually for each channel—by means of Fourier transformation:

$$P^n_c(\omega_k) \backsimeq \left|\mathcal{F}\left[\tilde{a}^n_{c,s}\right](\omega_k)\right|^2, \quad c = 1, \ldots, C.$$

The final feature values are computed by summation of the square root of the periodogram approximation within a specified frequency band (from ω_{low} to ω_{high}):

$$x^{\mathrm{HG}}_{c,n} = \sum_{\omega_k=\omega_{\text{low}}}^{\omega_{\text{high}}} P^n_c(\omega_k)^{\frac{1}{2}}, \quad c = 1, \ldots, C,\; n = 1, \ldots, N_w.$$

In summary, high gamma (HG) feature calculation depends on the following parameters: window size S_w, shift parameter Δ_w and frequency band $(\omega_{\text{low}}, \omega_{\text{high}})$. Since the investigations will focus on frequencies from the high gamma (HG) band (approx. 60–200 Hz), the features will be referred to as HG features.

3.2. Channel selection: extended Davies–Bouldin index

Based on the classic DB index (Section 2.2.2), a routine is constructed that is better adapted to the specific requirements for HMM decoding [44]. In a first step, all feature sequences are separated into three equally long parts. The index matrices $\mathbf{R}$ are then computed individually for each segment (and channel). The rationale behind that strategy is to introduce a temporal component to the evaluation of class separability. As HMMs are dynamic classifiers (cf. Section 2.3.1), it can already be sufficient for correct classification if a feature separates a certain class combination for a specific time period within the whole data segment. Focusing on high separability across the entire sequence might be too superficial, particularly due to the fact that the investigated time segments are usually rather long. The computed matrices are averaged across the three time segments. Geometric mean is used here in order to bias averaging towards smaller values—these correspond to higher separability. From the resulting averages, the feature with the smallest element of $\mathbf{R}$ is selected for the final subset. This feature provides the best separation of one class to at least one other class, i. e. it provides information to optimally settle one class decision. In the next steps, for each of the possible class combinations (i, j), the feature with the minimal index R_{ij} is added to the final subset. Each of these features optimally separates one specific class combination. Additionally, for each class, the feature with the smallest geometric mean of all R_{ij} is selected. This sums up to a total of

$$\underbrace{1}_{\text{lowest overall } R_{ij}} + \underbrace{(K^2 - K)}_{\text{best class separation}} + \underbrace{K}_{\text{best mean}} = 1 + K^2$$

features, provided that no feature appears in more than one of the stages (otherwise, only one occurrence is used). If this amount is smaller than the requested number of features, all remaining slots are filled up according to the average across all class combinations (lowest is selected first). Should there already be more features in the subset than requested, the last added features are removed until the count matches.

3.3. Model initialization

Before model parameters can be re-estimated using training samples, initial values are required for all parameters in the models. With respect to the initialization strategy, two different types of parameters can be distinguished: *a priori* defined and data-driven initialization. In principle, all parameters could be determined using some sort of approximation based on the training data. However, for some model parameters it is usually more meaningful to use a fixed initialization that is based on certain assumptions on the data.

3.3.1. Prior and transition matrix

Here, the prior $\boldsymbol{\pi}$ as well as the transition matrix $\mathbf{A}$ are initialized based on *a priori* definitions. Two different strategies for the prior are used: equal likelihood for each state ($\boldsymbol{\pi}^{\text{equal}}$, Section 4.1 only) and a prior that forces the model to start in the first state ($\boldsymbol{\pi}^{\text{first}}$, all other sections):

$$\begin{aligned}\boldsymbol{\pi} &= (\pi_i) \in [0,1]^Q \\ \pi_i^{\text{equal}} &:= \frac{1}{Q}, \forall i = 1, ..., Q \\ \pi_i^{\text{first}} &:= \begin{cases} 1 & i = 1 \\ 0 & i = 2, ..., Q \end{cases}.\end{aligned}$$

For construction of the transition matrices, all non-restricted elements are filled with ones. Additionally, the diagonal entries (i. e. state loop probabilities) are incremented by a term that is related to the length of the feature sequences S and the number of hidden states Q by $c := S/Q$. The rationale behind this specific design of the transition matrices is to consider that (on average) the model needs to dwell in its states for a number of times that is proportional to the sequence length in order to not end up in the final state too early[1]. The rows of the resulting matrix are individually normalized to fulfill the condition $\sum_j a_{ij} = 1, \forall i = 1, ..., Q$.

The following example shows the result for a four-state model that allows for single- and two-step forward-jumps (Bakis topology) when using features with $S = 28$ sample points ($c = 28/4 = 7$):

$$\tilde{\mathbf{A}} = \begin{pmatrix} 1+c & 1 & 1 & 0 \\ 0 & 1+c & 1 & 1 \\ 0 & 0 & 1+c & 1 \\ 0 & 0 & 0 & 1+c \end{pmatrix} = \begin{pmatrix} 8 & 1 & 1 & 0 \\ 0 & 8 & 1 & 1 \\ 0 & 0 & 8 & 1 \\ 0 & 0 & 0 & 8 \end{pmatrix}$$

$$\Rightarrow \mathbf{A} = \begin{pmatrix} 0.80 & 0.10 & 0.10 & 0 \\ 0 & 0.80 & 0.10 & 0.10 \\ 0 & 0 & 0.\overline{88} & 0.\overline{11} \\ 0 & 0 & 0 & 1.00 \end{pmatrix}.$$

3.3.2. Mean and covariances

State mean vectors $\boldsymbol{\mu}$ and covariance matrices $\boldsymbol{\Sigma}$ cannot reasonably be chosen based on *a priori* assumptions as they directly depend on properties of the acquired data. Hence, they are determined on the basis of a data-driven approach. The initialization technique used here is based on the well-known k-means clustering algorithm [110].

k-means clustering This routine aims at clustering data samples into a set of k clusters κ_i such that the total distance D of all J individual data samples x_j to the cluster means μ_i is minimized:

$$D = \sum_{i=1}^{k} \sum_{x_j \in \kappa_i} \|x_j - \mu_i\|^2 .$$

Computation of D requires an assignment of all data samples x_j to one of the clusters κ_i each. The usual procedure of finding the optimal clustering pattern to minimize D is an iterative refinement strategy, known as Lloyd's algorithm. This routine is a two-step strategy that consists of an *assignment step* (re-assign x_j to κ_i) and an *update step* (recompute means μ_i). In the t-th iteration, the individual steps are as follows:

Assignment step All data points are assigned to the cluster from which they have the smallest (squared Euclidean) distance:

[1]This is mainly true for left-to-right topologies, i. e. without backward jumps, and under the assumption of a prior that forces a start in the first state (π_i^{first}).

$$\kappa_i^{(t)} = \left\{ x_j : \left\| x_j - \mu_i^{(t)} \right\|^2 \leq \left\| x_j - \mu_{i^*}^{(t)} \right\|^2, \forall i^* = 1, ..., k \right\}.$$

Update step Based on the new clustering, cluster means are recomputed:

$$\mu_i^{(t+1)} = \frac{1}{N_i^{(t)}} \sum_{x_j \in \kappa_i^{(t)}} x_j,$$

whereby $N_i^{(t)}$ denotes the number of data samples that currently (i. e. in iteration t) belong to the cluster κ_i.

These steps are iteratively repeated until the assignments no longer change. Initial values for the cluster means $\mu_i^{(1)}$ need to be defined before the first iteration of the routine can be performed. A common strategy is to randomly select a data point from the set for each of the clusters to form its initial mean:

$$\mu_i^{(1)} = x_\xi,\ \xi = \mathrm{randi}(1, J),$$

with randi(a, b) denoting a pseudo-random number generator that returns an integer from the interval $[a, b]$. A drawback of this is the rather strong influence of the random initialization. This is due to the fact that different values for the initial cluster means can lead to entirely different final clustering results (depending on the actual data).

After clustering is finished, model parameters can be computed from the results. For initialization of a Q-state HMM, one sets $k = Q$. Each of the clusters serves as database for one of the HMM states. Mean and covariance parameters can be computed from the set of data samples that belongs to the clusters. Note that in such a setting, it is arbitrary which of the clusters gets assigned to which HMM state, as there is no information that could be used to decide this appropriately. This is addressed with the extensions presented in the following section.

3.3.3. Time discriminative *k*-means

To better adapt the initialization routine to the requirements for appropriate model parameters in HMM decoding, it has been slightly extended. Since HMMs are dynamic classifiers, ensuring a meaningful temporal structure within the model states can play an important role. This objective is addressed here by appending the (scaled) feature sample index s as an additional dimension to the feature vector $\mathbf{x}$:

$$\mathbf{x} = (x_{c,s}) \in \mathbb{R}^{C \times S}$$

$$\mathbf{x}^{\mathrm{TDI}} := \begin{pmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_S \\ \tau \cdot 1 & \tau \cdot 2 & \cdots & \tau \cdot S \end{pmatrix} = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,S} \\ \vdots & & \ddots & \vdots \\ x_{C,1} & & \cdots & x_{C,S} \\ \tau \cdot 1 & \tau \cdot 2 & \cdots & \tau \cdot S \end{pmatrix} \in \mathbb{R}^{(C+1) \times S}.$$

The time-coupling factor τ is used to control how strong the influence of the time dimension on the clustering result shall be. The higher τ is chosen, the more emphasis is put on the time point of a sample within the clustering process. Like in the conventional k-means routine, the final cluster assignments are used to compute the initial parameters for the HMMs. However, while clusters are arbitrarily assigned to a certain model state in the conventional approach, the time dimension in the feature vectors can be used here to assess a representative time point for the entire cluster. To do so, the average across the last feature dimension (i. e. the scaled sample index) is computed for each of the clusters. Clusters are then sorted with respect to increasing representative time point. Finally, the cluster with the earliest average time is assigned to the first state in the HMM, followed by the second earliest time point for state two and so on. That means that early HMM states are initialized based on data that (on average) comes from early time points in the feature sequence, whereas data from later parts of the sequence is used for the higher states. In doing so, a reasonable temporal structure is ensured.[2] After clustering, the appended time dimension is removed again from the feature vectors.

An appropriate choice for the time-coupling factor τ is particularly important if identical feature values can occur at different time points (which usually is the case). In case of clustering without respect to the time component (i. e. $\tau = 0$), those samples would all be assigned to the same cluster (since they have the same distance to it). As a consequence, they all end up in the data pool for initialization of the same Markov state for an HMM (Figure 3.1, 'Standard k-means'). Given that dynamic classifiers shall model the time course without any discontinuities, such a clustering would be irrational. This situation can be prevented with higher time-coupling factors, i. e. $\tau \gg 0$. However, arbitrarily high values of τ can reverse the situation and lead to another undesired outcome. If too much emphasis is put on the time dimension, the final clustering result corresponds only to the time domain, i. e. a cluster is always formed by consecutive samples of a time segment. Should there be substantial differences in the actual feature values within this segment, unrepresentative parameter values (i. e. state mean and esp. covariance matrix) can be the result (cf. example in Figure 3.1, 'TDI k-means ($\tau \to \infty$)').

3.4. n-gram models

Different strategies can be used to incorporate prior knowledge (PK) into the decoding, depending mainly on the specific type of available PK. In this work, PK incorporation is investigated in a setting that provides information on the frequency of certain event sequences.[3] The usual approach to consider this type of information is n-gram modeling. An n-gram is a sequence of n consecutive elements (e. g. phonemes or words). The n-gram model is a probabilistic model that uses the information on typical frequencies of appearance of the individual n-grams to make a prediction of the next event.

Here, bi-gram models are used to integrate *a priori* information in the context of a finger

[2]Note that this assumes that a left-to-right topology is used to connect the states.

[3]This is very similar to the situation in speech decoding, in which *a priori* knowledge on the frequency of appearance of phoneme combinations is available.

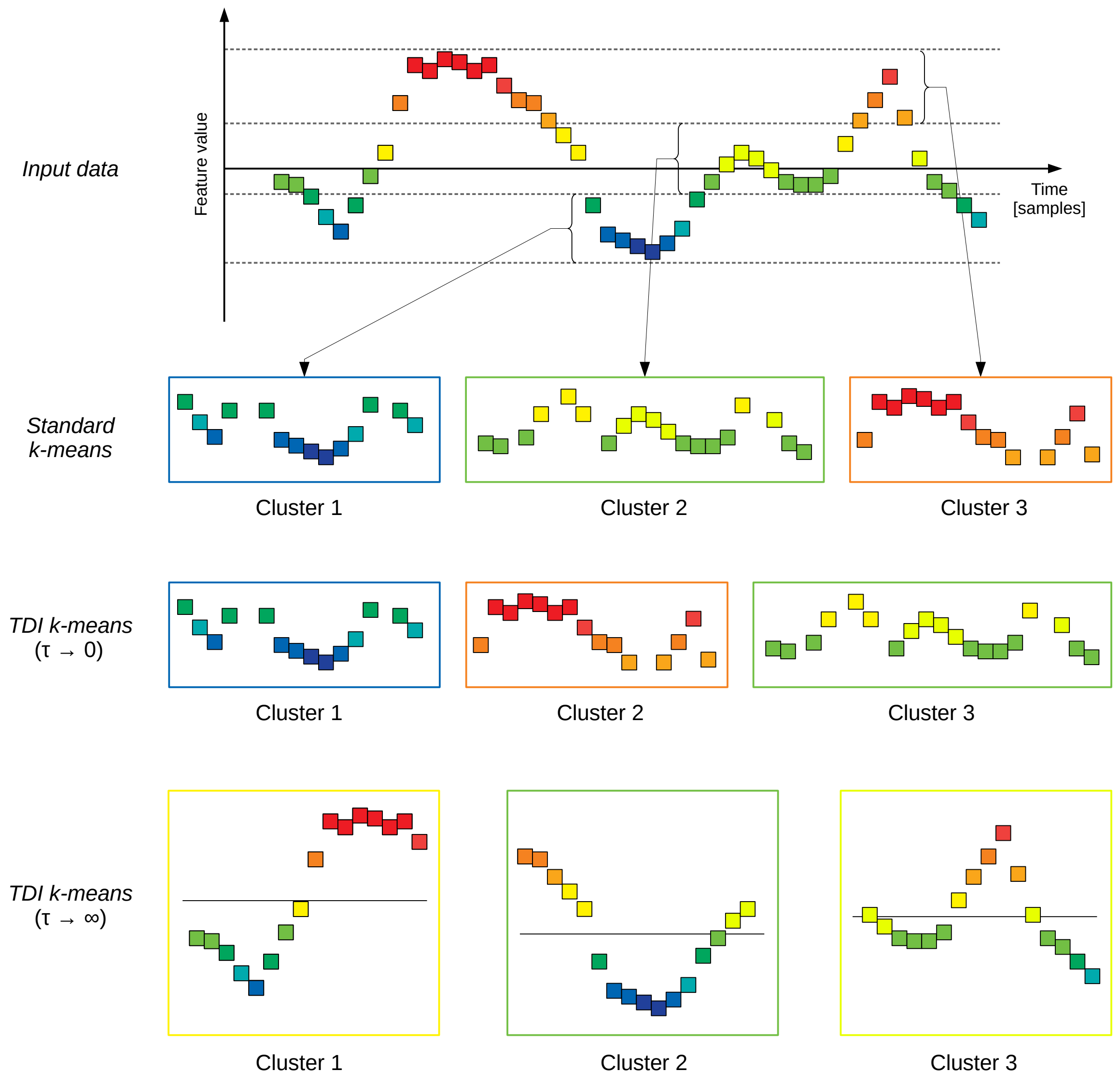


Figure 3.1: Illustration of k-means clustering results with different settings. For means of simplicity, input data of a single sequence with only one feature dimension is shown. The conventional k-means algorithm (top row) clusters feature samples that have similar values irrespective of their temporal position within the sequence. Contrary, the TDI k-means routine with very high time-coupling factor τ (bottom row) does not take into account the feature value at all, but simply clusters consecutive segments of data. Note that this leads to nearly identical mean values (indicated by black horizontal lines) for all three clusters, which potentially impairs modeling quality severely. Applying the TDI algorithm with a time-coupling close to zero (center row) leads to a similar clustering result as provided by the standard k-means routine. However, the temporal information is still considered by means of sorting the resulting clusters by their average time point of the contained data points. Any choice of $0 < \tau < \infty$ will result in some sort of compromise between the two shown TDI outcomes for the extreme cases.

movement experiment (Section 3.7.1). In the given experiment, prior knowledge is available on frequencies of pairs of consecutive finger movements (interleaved by rest). Consequently, bi-grams are used to represent these pairs of movements. A bi-gram model introduces the information on how often these movement combinations have been performed by the subjects. Full details on the routine can be found in Section 4.6.

3.5. Hidden Markov Model Toolkit / Software framework

The *Hidden Markov Model Toolkit* (HTK, [111]) counts to the most widely-used tools for automated speech recognition. It provides a broad variety of HMM-based decoding approaches. By concept, HTK is designed to meet the requirements for decoding of speech signals. Due to substantial structural differences between typical brain signals and audio data used in speech recognition, extensive adaptions are required to facilitate the use of HTK for a BCI context. This mainly refers to the following components:

- feature extraction routines,
- compatibility with multidimensional input,
- feature selection routines,
- model initialization.

Specification of all definitions and parameters required by HTK is quite tedious and can slow down the analysis workflow substantially. To provide a solution that allows for convenient investigation of relevant BCI-related problems, an all-in-one software framework has been developed as part of this work. The framework is implemented in C++ using a modular design as shown in Figure 3.2. All required sub-routines—this includes feature selection (Section 2.2.2) as well as model initialization (Section 3.3) methods—have also been implemented in this framework, except for feature extraction. Features are pre-calculated using existing Matlab implementations [44, 112, 113] and stored for later use in the HTK environment (imported by 'Data Browser'). To use these features in combination with HTK routines, they are converted into an HTK compatible format. The entire framework is connected to a graphical user interface (GUI) that has been developed to allow the user to conveniently manage data and define the settings. The framework constructs all files required by HTK and ultimately, runs the corresponding (pre-compiled) HTK tools. Prediction results are analyzed (and visualized) by evaluation routines as described in the following section. Examples of the usage of HTK routines, typical HTK files as well as screenshots of the actual user interface can be found in the Appendix A.3.

3.6. Evaluation methods

3.6.1. Single trial analysis: cross-validation

For single trial analysis, data are prepared as individual data segments—the so-called *trials*. Normally, each trial contains only a single event (e. g. a finger movement). The type of

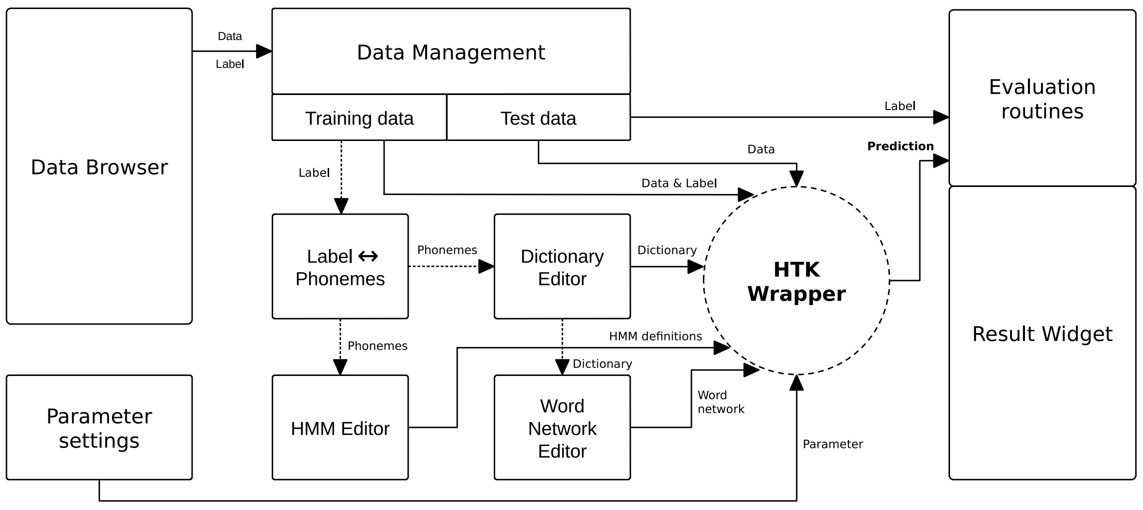


Figure 3.2: Structural overview of the developed software framework for HTK usage in BCI contexts.

the event determines the *class* the trial belongs to (e. g. index finger movements). The mapping of trials to the classes is called *labels*. The decoding task in a single trial setting is to determine the class membership of unknown trials using a trained classifier.

In many cases, data that shall be used for BCI studies are recorded and analyzed afterwards ('offline'). This holds true especially for ECoG data, since closed-loop scenarios are difficult to realize in the clinical environment, ECoG patients are in. For analysis, such data need to be split in separate test and training sets. However, this causes the result to be dependent on the actual assignment of trials to either be test or training data.

A very common method to perform single trial offline analysis in a well-structured manner is the so-called *cross-validation* (CV). The CV routine starts with splitting all data equally into a defined number of subsamples. The classifier is trained using data from all but one subsample. Decoding accuracy is then assessed on the one remaining subsample. This procedure is repeated such that each of the subsamples is used exactly once for evaluation (these repetitions are usually called *folds*). In doing so, it is guaranteed that the performance estimate is based on all available trials. To minimize the influence of the actual assignment of trials to certain subsamples on the performance estimate, the whole procedure is repeated multiple times with varying allocation of the trials. Finally, the average of all estimated performance values is taken. A CV routine with K subsamples that is repeated n times is usually referred to as n-times-K-fold CV.

It is not uncommon that the distribution of trials between the appearing classes in the available data is unequal, i. e. some classes have significantly more trials than others. This can be either due to inherent properties of the experimental paradigm (e. g. certain classes are simply less common) or it can be consequence of artifact removal that (coincidentally) affected some classes more than others. Straightforward application of the CV routine would likely yield unbalanced subsamples. In particular, this can result in a classifier bias towards classes with more training samples. In the worst case, one (or more) class would have no training samples at all, making it impossible to use such a subsample for training. Besides an unbiased training, it is usually desirable to estimate decoding accuracy on balanced test data; in doing so, the classifier needs to perform equally well on all appearing classes to achieve high performance values. To consider these aspects within evaluation, individual subsamples in the CV can be forced to contain equal numbers of trials from all classes (so-called 'balancing'). The downside of balancing is that some trials need to be discarded in each run in order to even out the class distribution. Hence, the total amount of training data is reduced and inter-run variances might increase.

3.6.2. Continuous results: pseudo-performance

Although single trial analysis is an extremely helpful tool to study basics of classification approaches and investigate various influences on accuracy of brain signal decoding, it is a quite artificial approach and it appears more natural to analyze continuous time series instead. This requires an individual prediction for each time sample in the data. Naturally, this significantly increases the complexity and thereby difficulty of the decoding task. While single trial decoding can be limited to just determine the type of an actual event, continuous decoding makes it necessary to also detect episodes of inactivity. These are commonly referred to as resting state.[4] Adequate modeling and detection of rest episodes in brain signals is highly non-trivial. The primary reason for that is: the human brain never idles completely; background activity of a broad variety of neural processes is always superimposed to the task related data and therefore, also present during periods of 'inactivity'. Besides that, certain experimental setups can make it difficult to find a direct interpretation of inactivity.[5]

Let Ω denote the set of possible label values. The decoded prediction $\mathbf{y} = (y_s) \in \Omega^S$ needs to be compared to the ground-truth labels $\mathbf{l} = (l_s) \in \Omega^S$ to assess decoding accuracy $\mathrm{Acc}(\mathbf{y}|\mathbf{l}) \in [0, 1]$. A straightforward method is comparison on a sample-by-sample basis:

$$\mathrm{Acc}(\mathbf{y}|\mathbf{l}) = \frac{1}{S} \sum_{s=1}^{S} \delta_{y_s l_s}. \tag{3.1}$$

This approach evaluates every single decoded sample and accumulates all samples for which the prediction matches the actual label. Equation (3.1) can be extended by an offset O to compensate for time shifts between prediction and actual event:

$$\mathrm{Acc}(\mathbf{y}|\mathbf{l}) = \frac{1}{S} \sum_{s=1}^{S} \delta_{y_{s+O} l_s}. \tag{3.2}$$

Apparently, it holds true that $\mathrm{Acc}(\mathbf{y}|\mathbf{l}) \in [0, 1]$ for any prediction $\mathbf{y}$. $\mathrm{Acc}(\mathbf{y}|\mathbf{l}) = 1$ refers to the case that all predicted values are identical to the labels; fully erroneous predictions

[4]In speech decoding, the analogue thereof would be silence.

[5]As an example, consider a setting in which subjects look at (natural) pictures. Even during the absence of a picture, subjects are observing their environment, which—*a priori*—is not categorically different from pictures as is also contains objects, people etc.

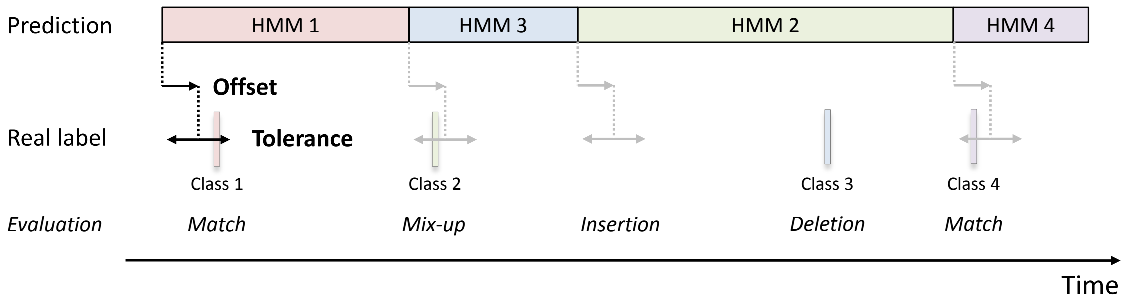


Figure 3.3: Evaluation of continuous data. Predictions are compared to ground truth labels within a certain tolerance window and under consideration of an offset between prediction and actual event.

yield $\mathrm{Acc}(\mathbf{y}|\mathbf{l}) = 0$.

Performance computation by means of Eq. (3.1) or (3.2) requires continuous label information $\mathbf{l} \in \Omega^S$. If this is unavailable, other approaches can be used to evaluate results based on the available information. Exemplary, a method is presented that compares (continuous) predictions to a set of labels containing only a time stamp (s_n) and the type (l_n) of each of the N events, i. e.: $\tilde{\mathbf{l}} = \{(s_n, l_n)\}_{n=1,...,N}$, $l_n \in \Omega$, $s_n \in \{1, ..., S\}$. Instead of comparing all individual samples to continuous label information, only the start samples of predicted events are compared to the real labels.[6] This eliminates the need for information on the actual duration of the events. Generally, predictions are unlikely to match the actual event at the exact sample point. To allow for small variations in timing, a *tolerance* window is specified within which the comparison is performed. The tolerance parameter controls the strictness of the evaluation. Higher values mean that an increased time window is considered to check for matching predictions. Naturally, this results in higher performance values. At the same time, results become less meaningful, as an increasingly higher error in the predicted time points of events is tolerated. Ideally, tolerance should be chosen as low as possible but high enough to allow for small timing fluctuations, which naturally appear due to variations in task execution and brain responses of different subjects.

As already introduced in the sample-by-sample method, an *offset* between prediction and event is also considered here (cf. Figure 3.3). The offset parameter is intended to compensate for potential delay between brain responses and actual events as well as for shifts that may be introduced artificially by signal processing steps.

Under consideration of offset and tolerance values, predictions are compared to the real labels with the following possible outcomes:

Match (M) Prediction matches real label within the tolerance time frame.

Substitution (σ) Prediction is at correct time point but with wrong class assignment.

Insertion (I) Prediction is at a time point without any actual event in the labels.

Deletion (D) Missing prediction at a time interval with an actual event in the labels.

[6] This is also referred to as an 'annotation task'.

Let N be the overall count of events in the data. Considering all possible decoding mistakes [98], the overall performance can be expressed as:

$$\mathrm{Perf} = 1 - \frac{\sigma + I + D}{N}. \tag{3.3}$$

Since every event in the data must be either classified correctly, substituted by another class or missed by the decoder, the overall count of events can be expressed as: $N = M + \sigma + D$. Equation (3.3) can then be rewritten as:

$$\mathrm{Perf} = \frac{N - \sigma - I - D}{N} = \frac{M - I}{N}.$$

Performance computed by means of Eq. (3.3) can have values from $-\infty < \beta \leq \mathrm{Perf} \leq 1$, with $\mathrm{Perf} = 1$ representing flawless decoding. Results with $\mathrm{Perf} = 0$ have approximately the same amount of events being detected correctly and those being inserted spuriously. In cases with a large number of insertions, performance values become negative. The lower bound β is given by the maximum amount of insertions that can occur in the decoded sequence. In the easiest case, this is simply the number of samples reduced by the count of real events in the data. When using an HMM decoder, the amount is further restricted by the topology of the model, as it may prohibit rapid changes between different models (and therefore, event classes).

The final procedure is as follows:

1. Translate the start sample of the prediction by the offset.
2. Within the specified tolerance window, compare prediction with real labels.
3. Compute overall accuracy under consideration of all decoding errors (Eq. (3.3)).

3.6.3. PK information estimate: entropy measure

In BCI experiments, the specifications of the paradigm determine how much prior knowledge (PK) is available that can be useful for decoding. To easily compare the benefit when incorporating PK into the decoding, it is desirable to quantify the amount of available PK by some sort of measure. Such a measure should ideally map information extent from 'none' (lack of any usable PK, e. g. in fully randomized settings) to 'fully deterministic' in a monotonous way.

As described in Section 3.4, PK incorporation is performed by means of bi-gram models in the context of this work. Hence, information content shall be quantified with respect to the capabilities of bi-grams. The central component to define the bi-gram model is the (relative) frequency of appearance $\mathbf{f} = (f_{ij})$ of transitions from one class to another. Let C denote the total number of classes. An entropy-based measure S can be computed from the frequencies $\mathbf{f}$ as follows:

$$S(\mathbf{f}) = -\sum_{i=1}^{C} S_i = -\sum_{i=1}^{C}\sum_{j=1}^{C} f_{ij} \cdot \ln f_{ij}. \tag{3.4}$$

In case of a sequence in which each subsequent event depends solely on the current one, i. e.

$$f_{ij}^{\text{det}} = \delta_{jj_i^*}, \forall i, j = 1, ..., C,\ j_i^* = 1, ..., C,$$

the resulting entropy value is $S(\mathbf{f}^{\text{det}}) = 0$. Consequently, the bi-gram model fully explains the entire sequence. For sequences containing higher degree of randomness, the entropy values will be larger. The upper limit for S is given by the entropy of a sequence with equal probabilities for all transitions, i. e.

$$f_{ij}^{\text{equal}} = \frac{1}{C}, \forall i, j = 1, ...C.$$

In that case, the upper limit $S^{\max} = S(\mathbf{f}^{\text{equal}})$ computes as

$$S^{\max}(C) = -\sum_{i=1}^{C} S_i = -\sum_{i=1}^{C}\sum_{j=1}^{C} \frac{1}{C} \cdot \ln\frac{1}{C} = -C^2 \cdot \frac{1}{C} \cdot (-\ln C) = C \cdot \ln C. \tag{3.5}$$

It is important to note that this measure can only reflect the degree of determinism—and by that the usable information for PK incorporation methods—that can be explained with a bi-gram model structure. Sequences can be fully deterministic with respect to higher order n-grams, but still have maximum entropy value (i. e. no usable information) when interpreted as bi-grams.[7]

3.7. Experimental paradigms

3.7.1. Finger tapping

In the *finger tapping study*, patients performed a serial reaction time experiment. A number was presented on a screen to indicate with which finger the patient should press a button on the keyboard. The numbers 1, 2, 3 and 5 represented thumb, index finger, middle finger and little finger, respectively. The ring finger was not included in the paradigm.[8] Throughout the experiment, patients' fingers rested on the standard touch typing home keys, i. e. 'space bar' (thumb) and 'ASDF' for left hand or 'JKL;' for right hand fingers. All patients used the hand on the contralateral side (i. e. on the opposite side) of the electrode grid location. Each patient was asked to respond as rapidly and accurately as possible. After any button press from the subject—whether correct or not—a new stimulus was automatically presented after a brief randomized delay of approximately 330 to 400 ms. Consequently, patients' response times controlled the overall finger tapping rate (i. e. number of button presses within a certain time interval); with quick responses resulting in higher frequencies.

The experiment was structured in three sub-block, each of which with a slightly different task:

[7] A simple three-class example can be found in the Appendix A.1.

[8] This is mainly due to many subjects having difficulties moving their ring finger individually from the other fingers.

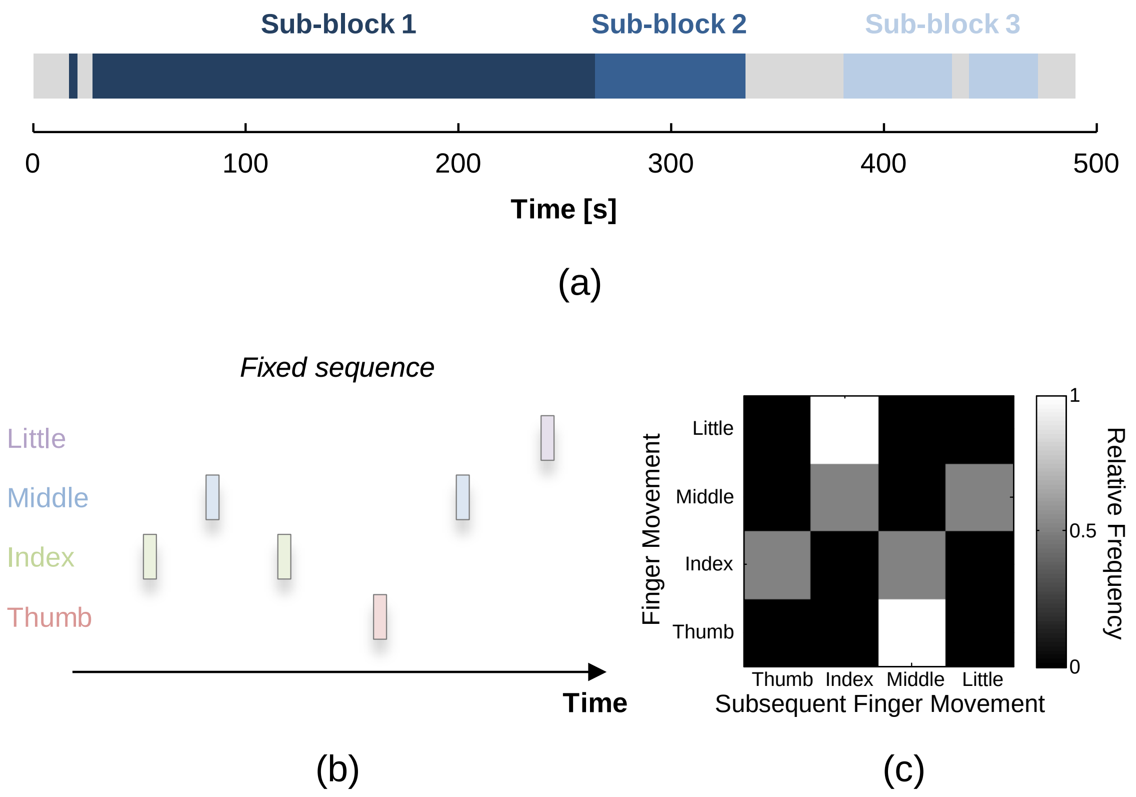


Figure 3.4: Overview of the finger tapping experiment. (a) General structure of a recording session. The three sub-blocks are interleaved with pause segments (gray color). (b) Structure of the fixed tapping sequence used in SB1. (c) Relative frequencies of occurrence of pairs of consecutive finger movements in SB1. (Figure reproduced from [46].)

Sub-block 1 (SB1) The first sub-block consists of a fixed sequence of stimuli that had been repeated 20 to 30 times. The fixed sequence was identical across all patients and sessions; it is illustrated in Figure 3.4 (b).

Sub-block 2 (SB2) In the second sub-block, random stimuli were presented to the patient. Randomization had been constrained such that consecutive stimuli could not have the same type. Consequently, out of the sixteen possible combinations of two consecutive stimuli, four appeared with a probability of zero.

Sub-block 3 (SB3) Within in last sub-block, patients could perform an arbitrary button press as response to the presentation of a stimulus. This freedom of choice significantly increased response times for most of the patients and hence, average tapping rates during SB3 were higher than in the other sub-blocks.

The finger tapping experiment provides data that are well-suited to study complex motor

BCI problems. Although overt movements[9] have been performed by the patients, high action rates (above one movement per second) and very brief motions (button press on a keyboard) present a challenging dataset. The different tasks across the sub-blocks offer the opportunity to study the possibilities of prior knowledge incorporation with various approaches.

ECoG data

Four patients—all of them received ECoG electrode implantation for pre-surgical planning of epilepsy treatment at UC San Francisco—volunteered to participate in the finger tapping experiment. All patients were right handed males and between 18 and 35 years old. Electrode grids were placed based on clinical criteria only. The implanted arrays covered various cortical areas, including pre-motor, motor, somatosensory, and temporal areas (electrode maps can be found in Figure 4.7).

Three patients were implanted with standard 8×8 ECoG grids containing 64 platinum–iridium electrodes with 1 cm center-to-center spacing. The electrodes in those grids had a diameter of 4 mm with 2.3 mm exposed. For one patient, a higher density grid with a total of 256 electrodes (16×16 array; 1.8 mm exposed, 4 mm spacing) was used. All electrocorticograms were recorded with a sampling frequency of 3051.7 Hz and down-sampled to 1017 Hz for storage. Signals have been pre-processed using a high-pass filter with a cut-off frequency at 0.5 Hz and notch filter around the power line frequency (60 Hz, USA). The measured electric potentials have been re-referenced to the common average.

Keyboard button presses were recorded simultaneously to the ECoG with the same sampling. There was no further monitoring of the patients' finger movements (like video recording, electromyography (EMG) or data gloves for position tracking). Therefore, detailed information on when a patient did or did not move the fingers is unavailable except for the time points at which a button is pressed.

3.7.2. Picture category task

This experiment consisted of a passive visual observation task. Patients looked at varying pictures that were presented to them on a screen. The pictures were taken from four different categories, namely objects, faces, watches and pieces of clothing. Stimuli were shown either on a projection screen (MEG data) positioned 1 m away from the subject or on a notebook screen (ECoG data) that was placed within the patients' reach. In order to keep up patients' attention throughout the experiment, they were asked to press a button each time a piece of clothing was presented. These trials (target stimuli) accounted for approximately 10 % of the total count of stimuli. In this work, data from the picture category task are intended for decoding of visual information only. As they include motor responses, target trials have been omitted for analysis to ensure that decoding is not biased by motor activity.

Measurements of brain activity were carried out in an ECoG study with two patients and in a supplementary MEG study with six healthy volunteers. Additionally, data from two of the MEG subjects had been re-recorded using a slightly modified paradigm (see below).

[9]Refer to Section 5.2 for a discussion on the relevance of overt movements.

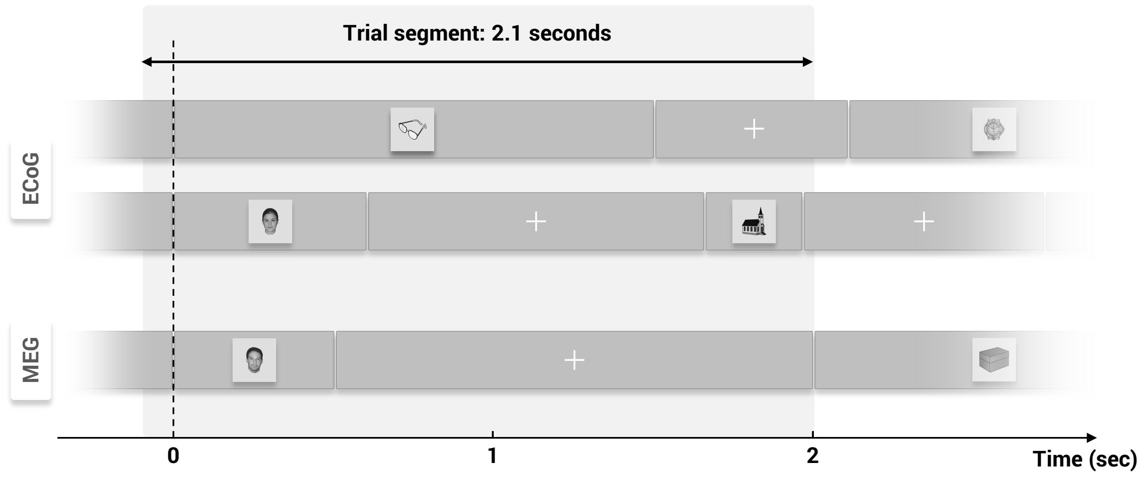


Figure 3.5: Scheme of the experimental paradigm in the picture category task. (Figure reproduced from [114].)

ECoG data Stimuli in the paradigm used for the ECoG study were presented for five different durations (300, 600, ..., 1500 ms). The inter-trial intervals also varied between five durations (600, 750, ..., 1200 ms).

The ECoG has been recorded from two patients (both right-handed males) for pre-surgical planning of epilepsy treatment at Stanford (CA, USA). Electrode grids of both patients had coverage of lateral occipital and medial temporal areas (early visual and fusiform area). Data have been sampled with 3051.7 Hz. Pre-processing steps include high-pass filtering with a cut-off at 0.5 Hz, application of a notch filter around the power line frequency of 60 Hz as well as re-referencing of all electrodes with respect to the common average.[10] For analysis, data were epoched into trials covering the interval from −100 to 2000 ms with respect to picture onset times ($t = 0$), totaling up to 2.1 s segment length.

MEG data Based on the identical experimental paradigm, MEG data have been recorded fro m six healthy subjects (age 23–31, one female). Recordings have been carried out with a sampling rate of 1017.25 Hz using a whole-head BTi Magnes system (4D-Neuroimaging, San Diego, CA, USA) equipped with 248 magnetometer sensors. For analysis, date were epoched in the same way as their ECoG counterpart (i. e. interval of [-100 2000] ms). All sensor signals were re-referenced in a bipolar manner, i. e. pair-wise differences were computed between neighboring sensors.

“Multi-stimulus issue” The randomization of picture presentation durations and inter-stimulus intervals resulted in combinations in which a new stimulus begins as early as 900 ms after the start of the previous one (duration 300 ms + ISI 600 ms). In order to ensure that stimuli with longer durations (e. g. 1500 ms) are fully covered, individual trials need to have

[10]The patients in this study have been implanted with several electrode grids (cf. example in Figure 4.14). Common average referencing has been carried out individually for each of these grids.

a correspondingly long segment length. As a consequence, some trials contain not only one stimulus at the beginning (i. e. the primary stimulus) but eventually also additional ones at the end (cf. Figure 3.5). This will be referred to as the *multi stimulus issue* (MSI).

MEG – modified paradigm In order to avoid the MSI, a slightly modified version of the paradigm has been developed for re-recording of MEG from two subjects (Section 4.3). Presentation durations have been extended in length such that the shortest duration is 500 ms (instead of 300 ms) and their selection has been reduced to three different values (500, 1000, and 1500 ms). Inter-stimulus intervals were prolonged significantly and have values of 1500, 1650, ..., and 2100 ms. In this constellation, the combination of the shortest presentation duration and ISI results in a time period of 2000 ms from the beginning of one stimulus to the beginning of the next one. Given that individual trials cover the time interval up to two seconds after stimulus onset, this ensured that no MSI could occur.

Data were recorded from a subset of the original MEG experiment's subject pool. This subset consists of two healthy, male subjects (age 26–28). All recording parameters (incl. acquisition setup, data epoching, and re-referencing) are identical to the original experiment.

4. Results

This chapter introduces all results of this work. The findings are structured in six sections, each of which contains a presentation of specific methods along with the actual results and a discussion of the approach. A detailed comparison of all results with related work and a thorough discussion of the limitations of the findings will be given in an overarching context in Chapter 5. The majority of results directly build up on each other, conceptually as well as methodologically. An illustration of relations between the individual studies is shown in Figure 4.1.

Sections 4.1–4.3 deal with the application of HMMs to single trial decoding tasks and the investigation of potential benefits of dynamic decoding approaches in such cases. In Sections 4.4 and 4.5, transition to continuous decoding is illustrated by the example of two different decoding scenarios. The last part (Section 4.6) is focused entirely on prior knowledge incorporation using HMMs in a continuous finger movement decoding problem.

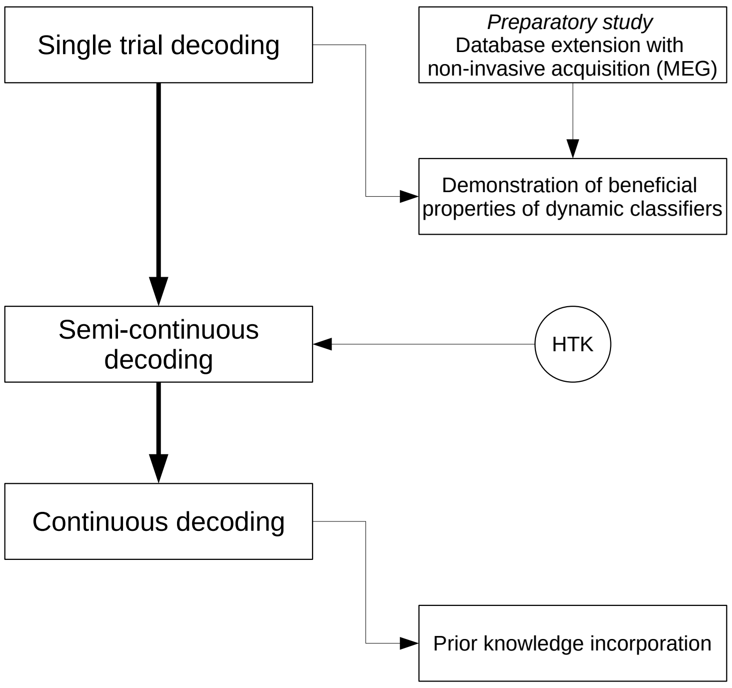

Single trial decoding
Preparatory study
Database extension with non-invasive acquisition (MEG)
Demonstration of beneficial properties of dynamic classifiers
Semi-continuous decoding
HTK
Continuous decoding
Prior knowledge incorporation

Figure 4.1: Overview of the result structure.

4.1. HMMs for (single trial) BCI decoding

The first result section focuses on the general applicability of HMMs in the context of BCI decoding and their comparison to gold-standard decoding routines. As mentioned earlier, HMMs are well-known from their application in ASR. Typical speech signals have properties that differ widely from those of brain signals. This makes it necessary to appropriately adapt the methods to the specific requirements of the measured signals. The most prominent difference is the presence of multiple electrodes in BCI recordings compared to an audio data stream, which is usually recorded by a single microphone. The count of electrodes, often also called *channels*, usually ranges from several tens up to a few hundreds. In most cases, only a fraction of these channels provide relevant information for a given problem. Careful selection of informative channels is of high importance for HMM-based decoding, as the count of free parameters that have to be estimated during training grows rapidly (at least quadratically, cf. Section 2.3.2) with the number of channels. Besides feature selection, suitable model topologies and appropriate strategies for model initialization need investigation. The ultimate goal of the investigations in this section is to demonstrate a prove-of-concept for the applicability of HMM decoders with comparable results as gold-standard static decoding routines in a single trial BCI classification context.

The central findings of this section have been published in [44].

Data ECoG data from the finger tapping experiment have been used in this study. Information on the experimental paradigm and technical details on the recordings can be found in Section 3.7.1.

Data were recorded from four subjects. Three of the subjects (S1, S3, and S4) participated in two sessions of data recording; subject S2 took part in four sessions. Patients' responses on the keyboard have been recorded simultaneously to the ECoG. The continuous data have been epoched into individual trials with respect to the button press times, covering an interval of two seconds (one second before and after the button press). The data were visually inspected for artifacts (e. g. epileptic spikes) and bad channels (e. g. improper electrode contact); affected trials have been rejected. Detailed statistics on the number of trials—including the count of rejected trials—are presented in Table 4.1.

Approach The performance of the HMM-based classifier shall be assessed for offline, single trial decoding of finger movement types. More precisely, it shall be predicted which finger was used to press a keyboard button. Each of the trials contains a single button press, which determines the label of the trial. Trials representing a *resting state*—that means blocks of data without any movements—are not considered here. This results in a four class decoding task.

Decoding is performed with individual HMMs for each of the classes (i. e. fingers). The models are constructed with five states and transition matrices are constrained to the Bakis model topology. This means that the model is restricted to state looping and forward jumps of one or two states (cf. Figure 2.4). The individual states are associated with a

Table 4.1: Trial statistics of all ECoG datasets from the finger tapping task. The first block ('Nb. of trials') lists all trials that have been used for analysis. A full breakdown on the number of rejected trials (identified by visual inspection) is shown in the second block ('Nb. of rejected trials').

Subject	Session	Nb. of trials					Nb. of rejected trials				
		All	THB	IDX	MID	LIL	All	THB	IDX	MID	LIL
S1	1	357	62	113	108	74	0	0	0	0	0
	2	240	33	90	72	45	1	0	1	0	0
	Total	**587**	**95**	**203**	**180**	**119**	**1**	**0**	**1**	**0**	**0**
S2	1	280	44	97	91	48	40	5	13	13	9
	2	308	51	103	111	43	0	0	0	0	0
	3	271	38	87	92	54	77	13	23	26	15
	4	273	32	88	88	65	41	8	17	10	6
	Total	**1122**	**165**	**365**	**382**	**210**	**158**	**26**	**53**	**49**	**30**
S3	1	234	54	70	70	40	114	16	40	36	22
	2	249	47	70	83	49	104	15	31	38	20
	Total	**483**	**101**	**140**	**153**	**89**	**218**	**31**	**71**	**74**	**42**
S4	1	359	57	113	116	73	6	2	2	1	1
	2	328	63	104	106	55	33	6	11	11	5
	Total	**687**	**120**	**217**	**222**	**128**	**39**	**8**	**13**	**12**	**6**

Abbr.: THB/IDX/MID/LIL – thumb/index finger/middle finger/little finger

single Gaussian mixture component and an unrestricted covariance matrix. To initialize the models, a modified version of the k-means clustering algorithm is used (see Section 3.3.3). In addition to the HMM classifier, decoding is also done using a linear support vector machine (SVM), as it represents a gold-standard reference routine in single trial BCI decoding. The SVM is used in a one-vs.-one setup. The penalty constant C is fixed to $C = 1000$. All routines (i. e. signal processing, decoding, analysis) are realized in Matlab 2012b.[1] HMM functionality is provided by the HMM toolkit for Matlab published by Kevin Murphy [115].[2] For SVM decoding, the LIBSVM package for Matlab [116] is used.

All decoding is done using a five-fold cross-validation (CV) routine (Section 3.6.1). Thirty repetitions of the CV routine are performed to minimize influence from the random assignment of trials to the individual folds. As seen in Table 4.1, trial counts are unequally distributed across the different classes. To avoid bias towards any of the classes that may arise from differing amount of training material, training data is balanced with respect to the count of trials of the classes, i. e. each CV set is guaranteed to contain equal amounts of trials of all the classes.

Two different types of features are investigated, namely low frequency time domain (LFTD) and high gamma (HG) features (see Section 3.1). All parameters that are re-

[1]MATLAB Release 2012b, The MathWorks, Inc., Natick, Massachusetts, United States.

[2]The HMM toolkit for Matlab is used in Sections 4.1–4.3. Starting with Section 4.4, the HTK framework is used instead (cf. Section 3.5).

Table 4.2: Parameter combinations in two-dimensional exhaustive search. All parameters apart from the parameter pair to optimize are fixed during optimization (Abbr.: ROI–region of interest).

Feature type	Parameter pair
LFTD	ROI (start & end)
	Cut-off frequency & down-sampling rate
HG	ROI (start & end)
	Frequency band (start & end)
	Window length & overlap

quired for extraction of the features are optimized using an exhaustive search approach (so-called *grid search*). Since the full parameter space is extremely high-dimensional, pairs of parameters have been optimized in separate two-dimensional grid searches with fixed values for all remaining parameters, based on empirical estimates.

Channel selection is performed using an algorithm based on the Davies-Bouldin index as described in Section 3.2. The ideal number of channels to be selected is determined using (one-dimensional) exhaustive search on the training subset.

Results Optimal parameter sets for all datasets are determined based on the two-dimensional grid-searches listed in Table 4.2. Throughout all optimizations, the number of channels is fixed to eight. The resulting optimal parameter values are shown in Figure 4.2. It becomes apparent that optimal values vary strongly across subjects and in some cases even across different sessions of the same subject. Particularly substantial variance is found for the ROI of HG features. The optimal value for the cut-off frequency used in LFTD feature extraction also varies strongly across subjects (11 Hz ... 30 Hz). However, within the same subject, almost identical optima are found—except for subject S2, which plays a special role (discussed later). Results for window size and offset of HG features are not shown, as these parameters turn out to have almost no influence on decoding accuracy. Hence, fixed values (across all datasets) are used instead. A window size of 260 ms is used. The window overlap is chosen such that the resulting number of feature samples in HG is equal to that of the corresponding LFTD feature.[3]

Using the determined parameters, final features have been extracted. Subsequently, the optimal number of channels is determined with one dimensional exhaustive search. The result of this optimization is shown exemplarily in Figure 4.3 for dataset S1-B1 when using HG features. To emphasize the effect of appropriate feature selection, corresponding results using the same amount of randomly selected channels are also shown. It can be clearly seen that decoding accuracy rises quickly to an optimum at around five channels. Contrary, adding in random channels provides only marginal benefits. Decoding accuracy with random channel selection raises slowly and reaches the decoding accuracy of the DB-index based selection not until all the channels are used. The peak performance—reached

[3]This is done for technical reasons, as it allows to combine both feature types (not shown here).

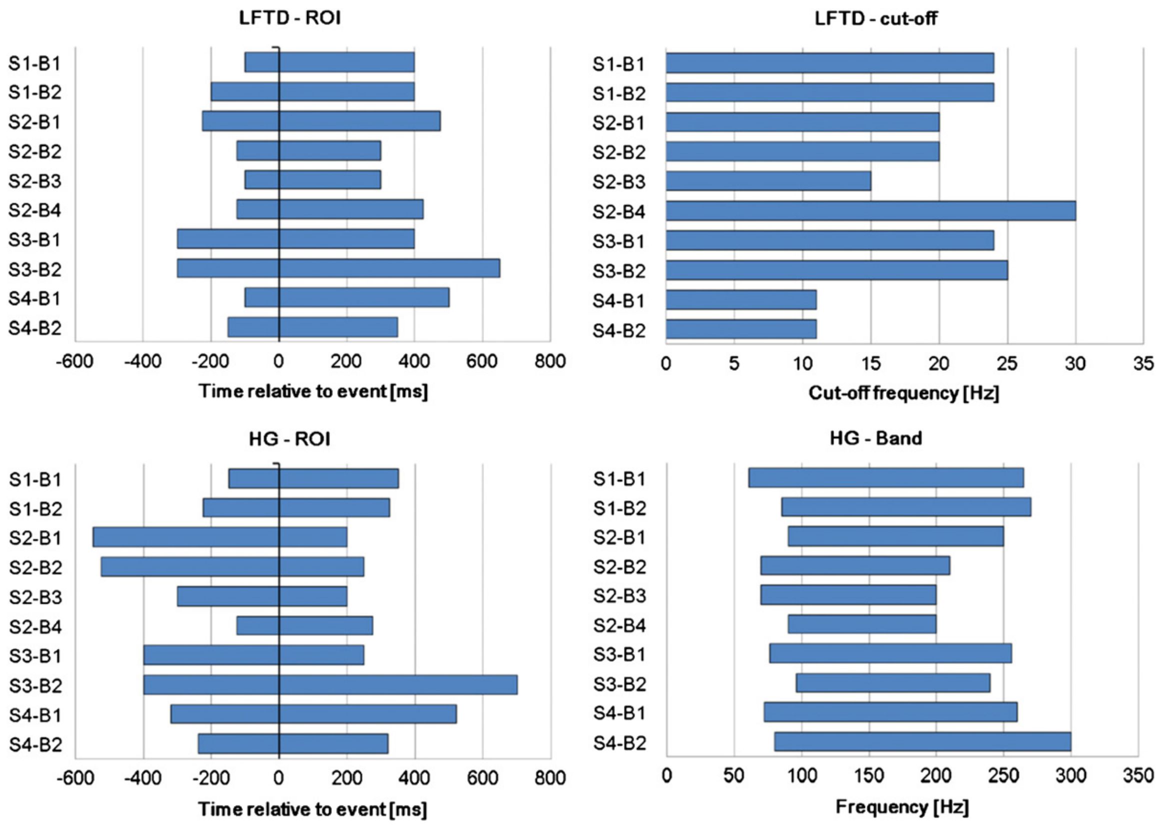


Figure 4.2: Optimal parameter values for all datasets. The results have been determined using two-dimensional exhaustive search with a fixed number of ten channels. (Figure reproduced from [44].)

when using the ideal number of channels—is about 6 % higher than the performance with all channels in use. This effect is even larger for LFTD features. Figure 4.4 exemplarily shows this for dataset S3-B2. In that case, the performance difference between optimal and full channel set is approximately 14 %. The ideal number of channels that results from optimization is listed in Table 4.3 for all subjects.

Figure 4.5 shows the decoding accuracies that are achieved with both classifiers for all datasets and both types of features. All results are averages across 30 repetitions of the 5-fold CV routine. The performance differences between all pairs of results have been analyzed for significance by means of two-sample one-sided t-tests (Table 4.4). In particular for LFTD features, SVM outperforms the HMM decoder in most of the cases (7 out of 10, all significant at $\alpha = 0.001$). Interestingly, the HMM decoder provides significantly better performance for subject S4. This also holds true for HG features, albeit at a lower level of significance ($\alpha = 0.05$). Overall, the superiority of SVMs becomes much smaller when it comes to HG features. Significantly better performances are mainly found for subject S2 (three sessions). For the other three subjects, performances of both classifiers are closely comparable. This is also reflected by the significance analysis (two non-significant differences; overall markedly

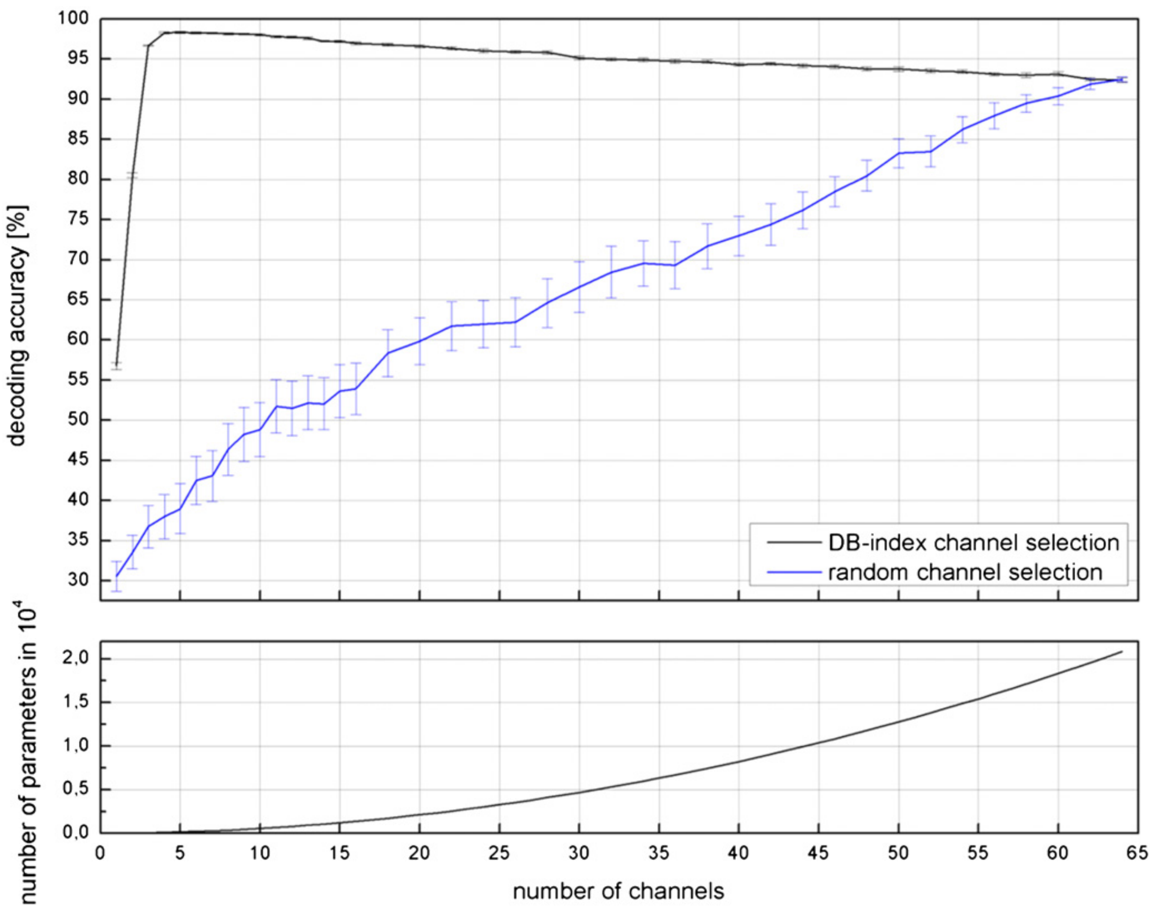


Figure 4.3: Decoding accuracy in dependence of the number of selected channels (top). Results are shown exemplarily for session 1 of subject S1 using HG features. Channel selection is carried out with an extended DB-index routine (Section 2.2.2). For comparison, decoding accuracies are also shown when channels are selected randomly (blue line). All decoding accuracies are assessed with 30-times-5-fold CV routines. The error bars indicate standard error.
(bottom) Total number of free parameters in the HMMs in dependence of the number of channels. (Figure reproduced from [44].)

Table 4.3: Average number of selected channels for all subjects. Channel counts are averaged across individual sessions (standard deviation in brackets).

Subject	LFTD	HG
S1	10.0 (0.0)	6.0 (0.0)
S2	6.5 (1.9)	7.3 (2.1)
S3	10.0 (0.0)	8.0 (0.0)
S4	10.0 (0.0)	6.0 (1.4)

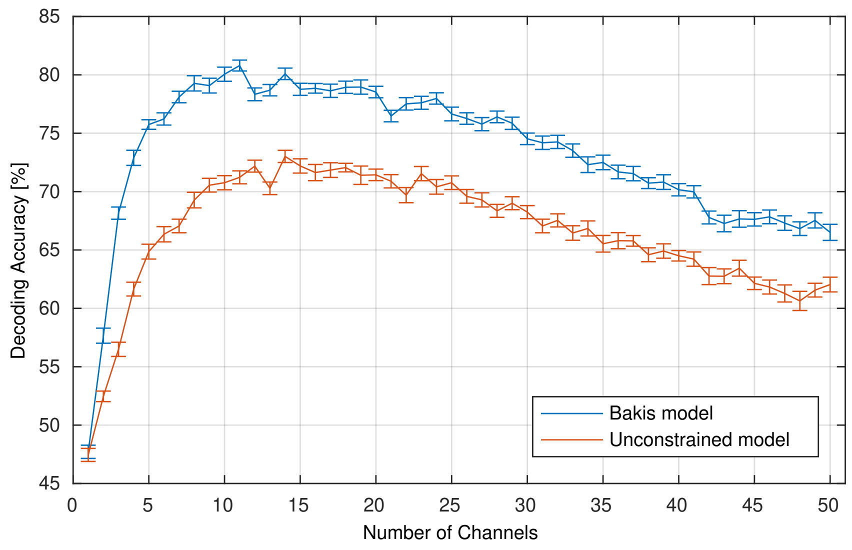
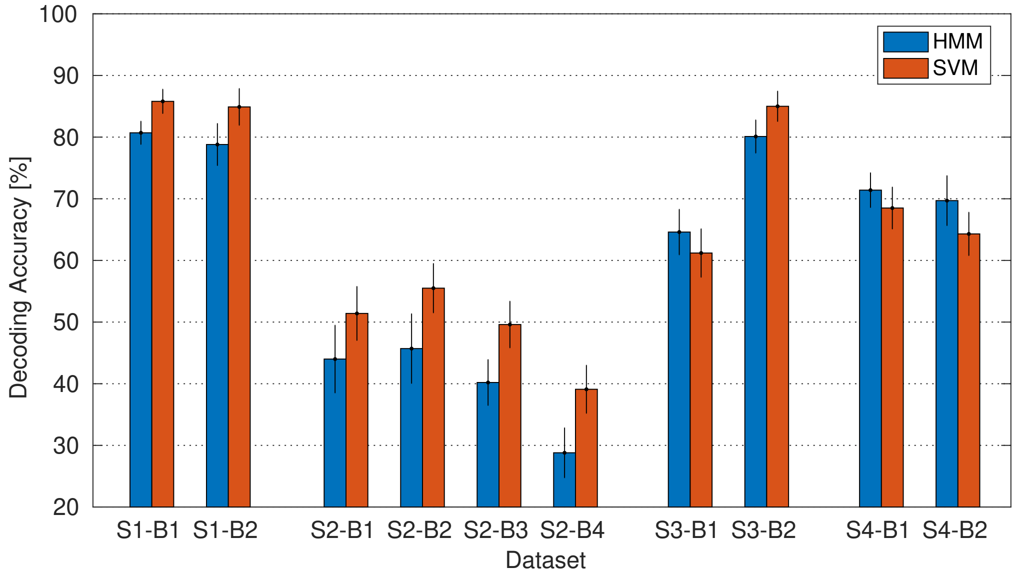


Figure 4.4: Decoding accuracy in dependence of the number of selected channels for dataset S3-B2 using LFTD features. Accuracy is shown for the decoder that is restricted to the Bakis topology (blue line) and for the unconstrained model (red line). For details on the approach, see caption of Figure 4.3. (Figure adapted from [44].)

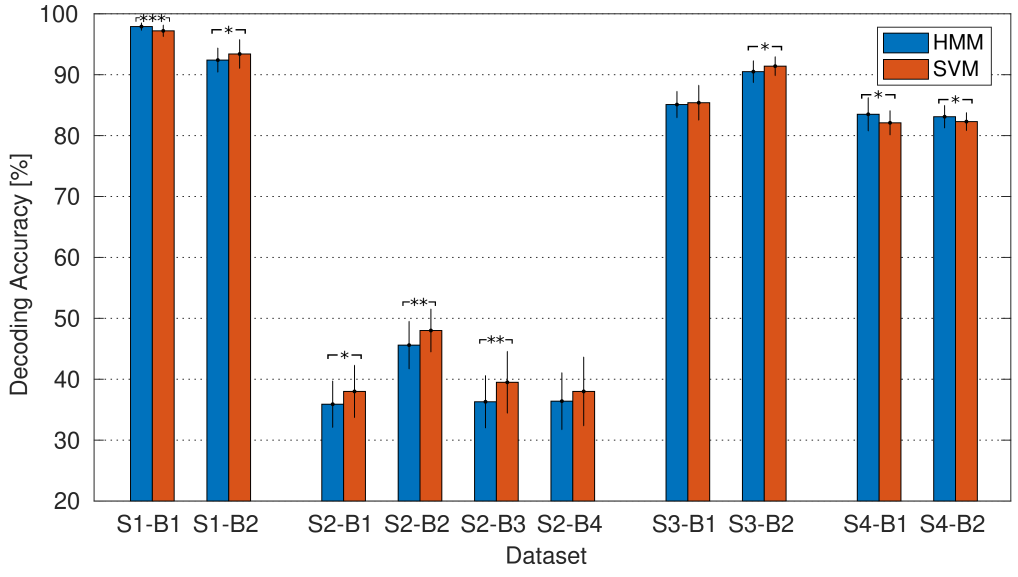
higher p-values). In addition to the decoding accuracies, confusion matrices for all subjects (averaged across individual sessions) are shown in Figure 4.7. The results show that—except for subject S2—a pair of finger movements can be identified for each subject that is more or less frequently mixed-up. For subject S1, this refers to index and little finger. In case of HG features, decoding errors occur almost exclusively in form of mix-ups between these two classes (5.2 % avg. error rate; average across all other mix-ups: 0.9 %). For subjects S3 and S4, a closely similar situation is observed for index and middle finger movements. Mix-ups between these classes make up a considerable amount of the total decoding errors (S3: 42 %/67 %, S4: 37 %/66 % of all errors for LFTD/HG). Contrary, for subject S2, decoding errors are distributed more equally, in particular for HG features.

In summary, it can be concluded that HMMs reach the performance level of SVM decoders in case of HG features for three of the subjects. Subject S2 seems to play a special role with respect to the achievable decoding accuracies as well as the distribution of decoding errors. This will be investigated more closely in the following.

Based on the selected channels in all repetitions (and folds) of the CV routine, maps can be created, showing precisely which electrodes on the grid provided the most useful information. To do so, the count of how often a specific channels was selected has been computed and normalized to the total number of times the selection routine was performed. This results in relative selection rates for all channels (Figure 4.7). It becomes apparent that for all subjects but subject S2, a 'cluster' of electrodes can be found that is selected almost always (i. e. selection rate ~100 %). Comparison of the location of these channels shows that they lay close to the central sulcus. However, for subject S2, hardly any of the channels is selected with particularly high rates. When it comes to HG features, not



(a) LFTD features



(b) HG features

Figure 4.5: Comparison of decoding accuracies between HMM and SVM decoder for all datasets. Results are shown for LFTD (a) and HG features (b). Decoding accuracies are averages from 30-times-5-fold CVs. The error bars indicate standard deviation across the 30 repetitions. Chance level for this four-class decoding problem is 25 %. Significance analysis has been carried out by means of two-sample one-sided t-tests. Star symbols above the pairs of bars show the corresponding level of significance (*/**/*** for $\alpha = 0.05/0.01/0.001$). Bars without indicator have non-significant difference (sub-figure (b) only). All results shown in sub-figure (a) have statistically significant performance differences at $\alpha = 0.001$ (***-symbols omitted).

Table 4.4: Detailed decoding accuracies (in %) of the HMM and SVM decoder for all datasets. All results are averages from 30-times-5-fold CVs (standard deviation in brackets). Significance analysis has been carried out by means of two-sample one-sided t-tests. The t-value of the test as well as the corresponding p-value are shown in the last two columns. For cases in which performance differences are statistically significant, the superior routine as well as the level of significance (*/**/*** for $\alpha = 0.05/0.01/0.001$) is listed.

Subject	Session	LFTD features				
		HMM	SVM	superior	t	p
S1	1	80.7 (1.9)	85.8 (2.0)	SVM (***)	-10.258	6.00E-15
	2	78.8 (3.4)	84.9 (3.0)	SVM (***)	-7.378	3.40E-10
S2	1	44.0 (5.5)	51.4 (4.4)	SVM (***)	-5.771	1.62E-07
	2	45.7 (5.7)	55.5 (4.0)	SVM (***)	-7.754	7.93E-11
	3	40.2 (3.7)	49.6 (3.8)	SVM (***)	-9.708	4.60E-14
	4	28.8 (4.1)	39.1 (3.9)	SVM (***)	-10.034	1.37E-14
S3	1	64.6 (3.7)	61.2 (3.9)	HMM (***)	3.454	5.19E-04
	2	80.1 (2.7)	85.0 (2.5)	SVM (***)	-7.363	3.60E-10
S4	1	71.4 (2.8)	68.5 (3.4)	HMM (***)	3.596	3.34E-04
	2	69.7 (4.1)	64.3 (3.5)	HMM (***)	5.519	4.17E-07
Average		60.4 (19.0)	64.53 (16.5)	SVM (*)†	-2.222†	0.027†

Subject	Session	HG features				
		HMM	SVM	superior	t	p
S1	1	97.9 (0.6)	97.2 (0.9)	HMM (***)	3.421	5.74E-04
	2	92.4 (2.0)	93.4 (2.4)	SVM (*)	-1.778	0.040
S2	1	35.9 (3.8)	38.0 (4.3)	SVM (*)	-2.005	0.025
	2	45.6 (3.9)	48.0 (3.5)	SVM (**)	-2.502	0.008
	3	36.3 (4.3)	39.5 (5.1)	SVM (**)	-2.636	0.005
	4	36.4 (4.7)	38.0 (5.6)	none	-1.197	0.118
S3	1	85.1 (2.2)	85.4 (2.9)	none	-0.458	0.324
	2	90.5 (1.8)	91.4 (1.6)	SVM (*)	-2.082	0.021
S4	1	83.5 (2.7)	82.1 (2.0)	HMM (*)	2.270	0.013
	2	83.1 (1.9)	82.3 (1.5)	HMM (*)	1.854	0.034
Average		68.7 (26.4)	69.5 (25.3)	none†	-1.799†	0.053†

Critical t-values: $t_c = \pm 1.672/\pm 2.392/\pm 3.237$ for $\alpha = 0.05/0.01/0.001$, respectively.

†Significance analysis for differences in the averages is carried out with one-sample one-sided t-tests on the performance differences $\mathrm{Acc}^{\mathrm{HMM}} - \mathrm{Acc}^{\mathrm{SVM}}$ (alternative hypothesis: $\mathrm{Acc}^{\mathrm{HMM}} - \mathrm{Acc}^{\mathrm{SVM}} < 0$) in the individual datasets. The critical t-value for these tests (significance level $\alpha = 0.05$) is $t_c = -1.812$.

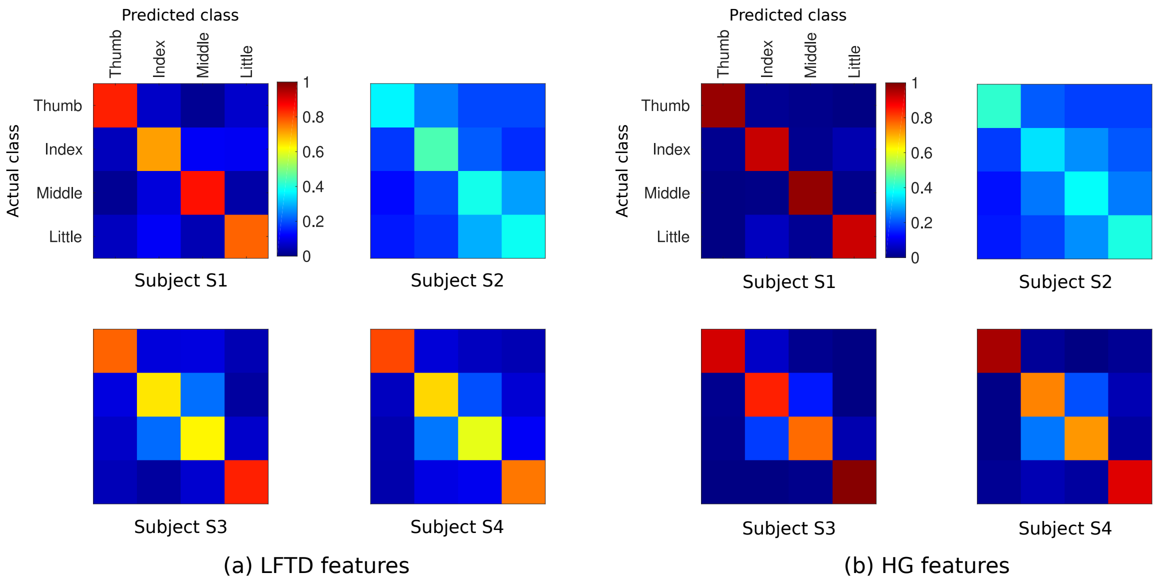


Figure 4.6: Confusion matrices of HMM decoding for both feature types. The confusion matrices show the relative distribution of decoding results. Rows correspond to the actual label and columns to the prediction from the classifier. The matrix elements (color-coded) show the fraction of classifier decisions that belong to the corresponding combination. Diagonal entries represent correct predictions (i. e. actual class = predicted class). Hence, in case of perfect decoding the confusion matrix would by the identity matrix. All shown results are averages across all individual sessions of a subject.

a single electrode is chosen by 100 %. Only three electrodes have selection rates of about 75 %, and all of those are located in the very corner of the grid. This strongly indicates that the electrode grid did not sufficiently cover the brain area associated with finger movement activity for that subject. Hence, data from this subject are not optimally suited for the purpose of finger movement decoding.

As introduced in the 'Approach' paragraph, the model has been constrained to the Bakis topology in order to increase performance by reducing the degrees of freedom. To investigate the benefit that results from these restrictions, decoding accuracies have also been assessed with an unconstrained model. The results of this comparison are shown in Figure 4.8. Except for subject S2, use of the Bakis model leads to an increase in decoding accuracy for all datasets. In most cases, these differences are statistically significant (detailed results—including significance test statistics—can be found in Appendix Table A.1). On average, the performance increase is higher for LFTD data ($\Delta\mathrm{Acc}^{\mathrm{LFTD}} = 2.2\,\%$ vs. $\Delta\mathrm{Acc}^{\mathrm{HG}} = 1.1\,\%$). Significance analysis verifies that only the increase in the LFTD case is statistically valid. This has been tested using one-sample one-sided (alternative hypothesis: $\Delta\mathrm{Acc}^{\mathrm{LFTD/HG}} > 0$) t-tests performed on the accuracy differences:

$$t^{\mathrm{LFTD}} = 2.665\,(t_{c,\alpha=0.05} = 1.812) \Rightarrow \text{significant at } \alpha = 0.05\,(p = 0.013)$$
$$t^{\mathrm{HG}} = 1.758\,(t_{c,\alpha=0.05} = 1.812) \Rightarrow \text{not significant at } \alpha = 0.05\,(p = 0.056).$$

To provide further insight, Table 4.5 lists the number of free parameters of the Bakis model

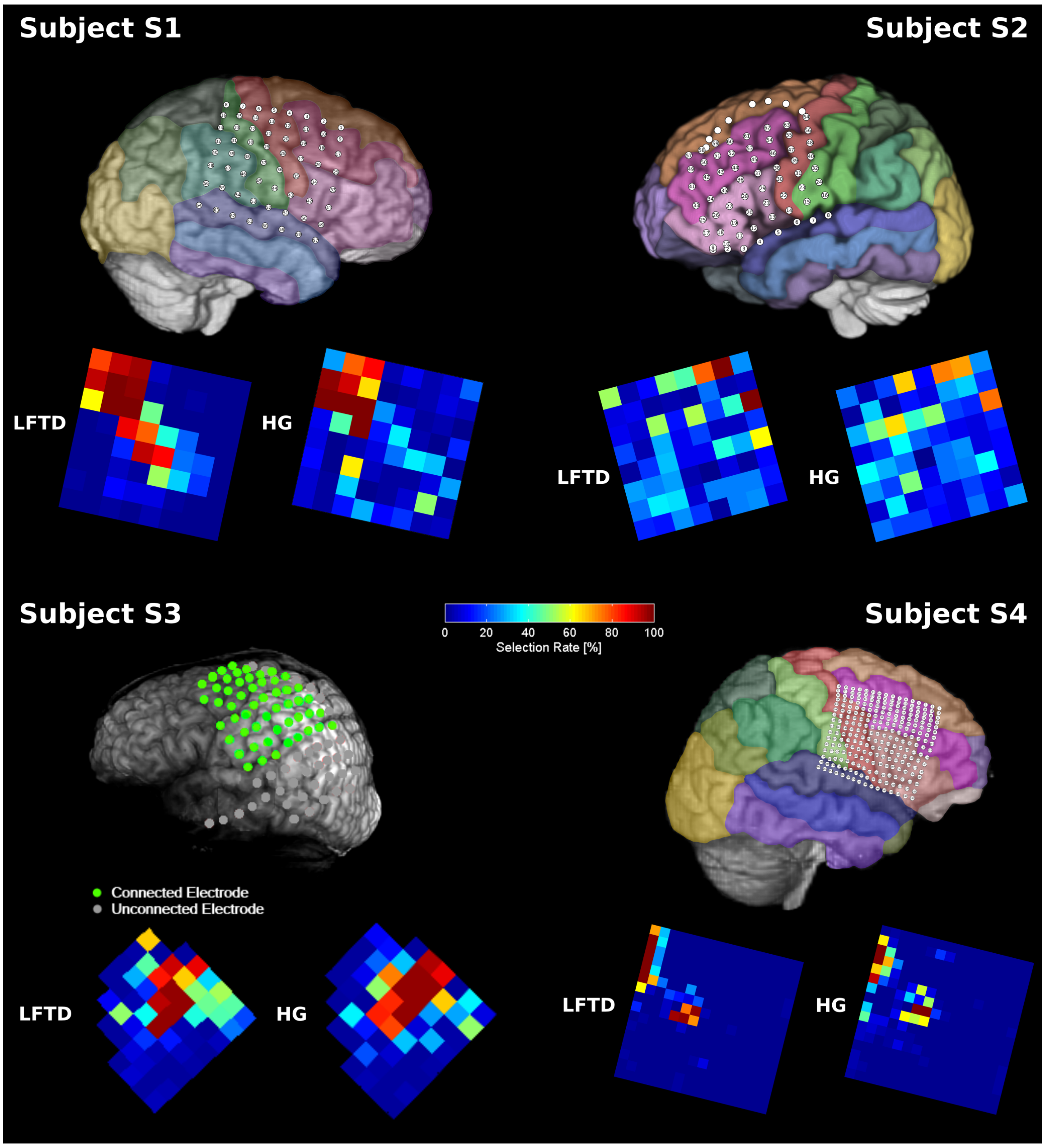


Figure 4.7: Spatial distribution of dominantly selected channels (channel maps). The results show how often (rate in %) channels have been selected throughout all runs and folds of the 30-times-5-fold CV routines. All maps show average selection rates across all individual sessions of the subject. For comparison, anatomical grid location for all subjects is shown above the corresponding channel maps. (Figure adapted from [44].)

Table 4.5: Number of free parameters in the unconstrained HMM and the model restricted to the Bakis topology.

Model type	Prior	Transition matrix	Mean	Covariance	Sum
unconstrained	5	25 (full)	$5 \cdot 10 = 50$	$5 \cdot (10 \cdot 10) = 500$	580
Bakis	5	12 (restricted)	50	500	567

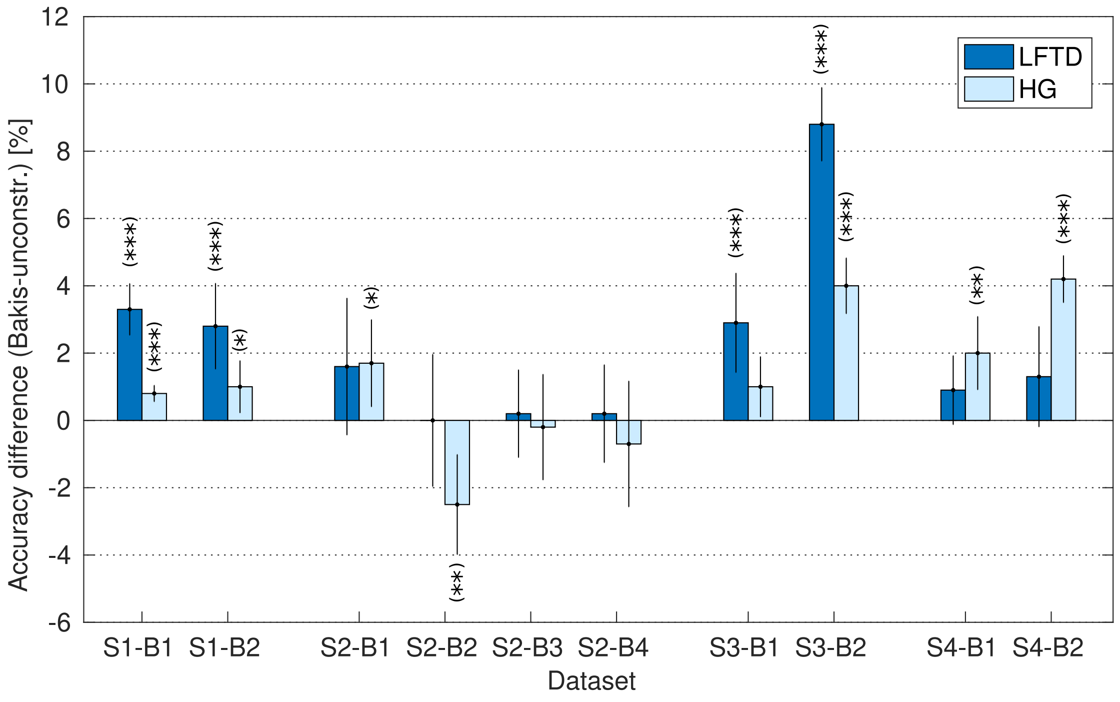


Figure 4.8: Difference in decoding accuracy between Bakis model and the unconstrained HMM for both feature types. Positive values indicate that the Bakis model provides higher decoding accuracy. All differences have been tested for statistical significance by means of two-sample one-sided t-tests (star symbols show the level of significance: */**/*** for $\alpha = 0.05/0.01/0.001$).

and the corresponding unconstrained HMM. Interestingly, the Bakis model reduces the amount of free parameters by only 13 values (roughly 2 %) and still provides significant performance benefits in a variety of cases. This indicates that restrictions to the possible state sequences play an important role to increase stability of the decoding results.

Discussion Detailed investigations of the behavior of the HMM classifier mainly revealed two important aspects. First, appropriate selection of features seems to play a crucial role for reliable decoder performance. Particularly for the lower quality LFTD features, substantial performance differences are found between the optimal and non-optimal channel counts. This can be attributed to the rapidly increasing number of free parameters (quadratic increase with number of channels) for HMM classifiers and the introduction of uninformative feature dimensions when channels are added that do not cover relevant brain areas. Second, introduction of proper model constraints is of high importance to improve the performance. Compared to a fully unconstrained setting, accuracy increases of up to 8.8 % (absolute increase, S3-B2 LFTD features) are achieved with the applied Bakis model topology. In that respect, it is particularly interesting that the constraints introduced by the Bakis topology cause almost no reduction in the total number of free parameters (Table 4.5). Still, significant performance increase is observed in almost all cases (Figure 4.8). This strongly suggests that the improvements cannot be simply attributed to a mere reduction in the overall number of parameters, but are also due to better compliance of the model topology with the underlying temporal structure of the features. Further support for this hypothesis comes from the fact that the performance benefit of the Bakis constraint is markedly higher for LFTD features, which exhibit the more complex temporal structure.

The decoding results do not show clear superiority of either HMM or the (gold-standard) SVM classifier for subjects S1 and S3. Both methods provide (significantly) higher performances in individual cases. Clear dominance of the SVM decoder is found for data of subject S2. Across all sessions and both feature types (except for S2-B4 HG), SVM results are significantly higher than those of the HMM routine. However, as already indicated in the results, subject S2 data play a special role. In all settings, decoding accuracies are substantially worse than those of all other datasets. This becomes particularly obvious in case of HG features, for which an average decoding accuracy of only ~40 % is reached for subject S2, while all other results have accuracies above 80 %. Non-optimal grid placement for the purpose of finger movement decoding has been identified as a likely explanation for the poor decoding results in these datasets. Channel maps show unspecific distribution of selected channels across the electrode grid of subject S2. For HG features, none of the channels is selected in all cases. The complete opposite is observed for the three other subjects. Distinct clusters of dominantly selected channels—with several channels selected by 100 %—can be identified for both feature types in all datasets. Moreover, subject S2 data is the only case in which decoding accuracies are higher for LFTD features than for HG features. Since high gamma signals are known to be spatially more focused [117–119], this strongly indicates that brain regions with relevant motor activity are located outside the electrode grid for subject S2. Hence, actual grid coverage does not allow for meaningful decoding of finger movements and thus, explains the fundamental differences in the results compared to the datasets of the other subjects.

In case of subject S4, the HMM decoder outperforms SVMs. For both feature types and data sessions, decoding accuracies of the HMM routine are significantly higher than the SVM performance. This also holds true in case of LFTD features, for which the SVM decoder is clearly superior in almost all other datasets. Interestingly, this subject has been implanted with an electrode grid with higher resolution (cf. Figure 4.7). This is an indication that HMM decoder might have higher benefit from increased spatial resolution than SVMs.

In general, results show that the performance gap between HMM and SVM gets smaller the higher the overall decoding accuracy becomes. Hence, findings suggest that for high quality data, both classifiers tend to perform equally well. Considering that HMMs—as a representative of dynamic, probabilistic classifiers—offer several interesting features for potential extensions that might turn the results in favor of HMMs, this is a promising observation. As exemplary advantages, the fact that HMMs directly model the time sequence of the data, which might be used to assess additional information (dynamic classification, investigated in Section 4.3), as well as the possibility to easily incorporate prior knowledge into the decoding (due to probabilistic classification, investigated in Section 4.6), shall be mentioned here.

Conclusion The findings of this section demonstrate that HMM decoders can reach the performance of one of the gold-standard static classifiers—the SVMs. This holds true especially for high quality data and high decoding accuracies. In summary, the results strongly motivate further analysis of the possibilities of HMMs in more complex scenarios, which better comply with the intended field of application for this type of decoder than a straightforward single trial classification task. This will be addressed in the following sections.

Comparison of the results of individual subjects showed that the data of one of the subjects (S2) is unsuited for meaningful decoding of finger movements, most likely due to unfitting electrode grid placement and the insufficient data quality that is caused by that. Consequently, data from this subject will be left out for further investigations (Sections 4.5 and 4.6).

4.2. Comparison MEG-ECoG

Typically, ECoG studies are conducted using data from only a few patients. This is an inevitable consequence of the limited patient collective that undergoes grid implantation solely for medical reasons. Furthermore, grid location is determined based on clinical needs and hence, coverage of brain regions varies across patients. Consequently, not every subject can participate in a particular experiment and the amount of available data is reduced even further.

To conclusively investigate the generalizability of results, it is desirable to evaluate decoding algorithms on as many (suitable) datasets as possible. Non-invasive acquisition modalities offer the possibility to record data from healthy subjects. However, significant differences in data properties are to be expected. If such data shall be used to extend ECoG databases, it is necessary to carefully investigate the data properties for their suitability for the problem of interest.

This aspect plays an important role for the analysis of ECoG data from the *picture category task* (Section 3.7.2) that will be presented in Section 4.3. Compared to the finger tapping data with a total of ten datasets (four patients, multiple sessions each), the database for the picture category task is particularly limited (two patients, only a single session each). Meaningful analysis of brain signals from a visual task requires data recorded from brain areas that play a central role in the processing of visual information (primarily occipital and medial temporal lobe). Due to the rareness of epilepsy located in these areas of the cortex, the corresponding patient collective is extremely limited. Hence, only very few datasets are available for BCI studies. As mentioned before, non-invasive data acquisition constitutes a potential candidate for database extension. This section deals with a brief comparison of data properties, conducted to clarify if the non-invasively recorded data is suited for database extension in the analysis of the picture category task.

Data In addition to the ECoG data, MEG data have been recorded using the same experimental paradigm as in the ECoG picture category task (Section 3.7.2). Six healthy subjects (age 23–31, one female) participated in the study. Data were recorded with a sampling rate of 1017.25 Hz using a whole-head BTi Magnes system (4D-Neuroimaging, San Diego, CA, USA) equipped with 248 magnetometer sensors.

Approach In a first step, MEG and ECoG raw signals as well as extracted features are inspected visually to compare essential properties. Both, LFTD and HG features (Sections 2.2.1 and 4.1) are considered. LFTD cutoff frequency is 30 Hz for ECoG and 10 Hz for MEG data. HG features are computed with a window length of 250 ms) in the frequency band of 70–200 Hz. Feature selection by means of the extended DB-index routine (Section 2.2.2) is performed to identify the most informative channels (ECoG: electrodes, MEG: magnetometer sensors).

To obtain quantitative statements on the information content of the signals, decoding accuracies are assessed for both data types.

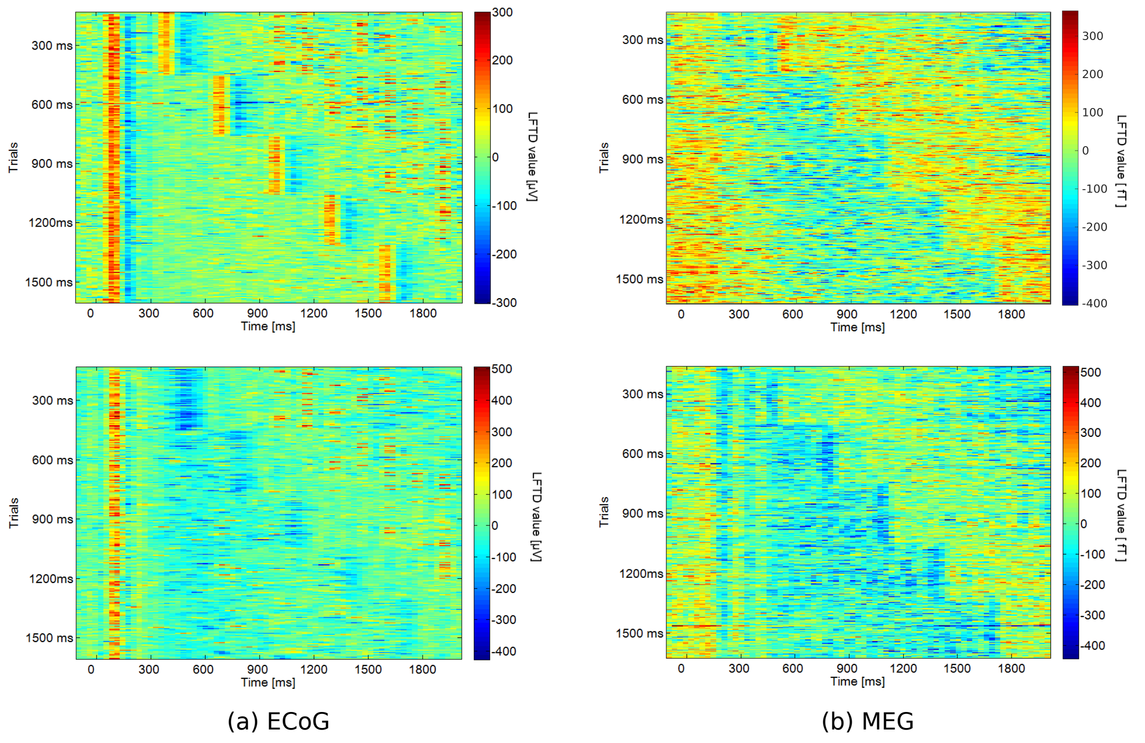


Figure 4.9: LFTD feature values of two representative channels from ECoG (a) and MEG (b) data. The plots show the LFTD feature value (color-coded) for all individual trials sorted with respect to the stimulus presentation duration (shortest duration on top). Sub-figure (a) shows the two top-ranked (w. r. t. feature selection index from the extended DB-routine) channels from subject ECoG 2. In sub-figure (b), the top-ranked channels from two MEG datasets—MEG4 (top) and MEG1 (bottom)—are shown for comparison. (Parts of this figure are reproduced from [120].)

Decoding is conducted for both possible decoding tasks:

1. Identification of the type of a stimulus (i. e. picture category) and
2. determining the stimulus duration.

To obtain unbiased results, a gold-standard (linear) SVM approach is used to perform classification. The feature selection routine is used to identify the twenty (MEG: ten) most informative channels with respect to the decoding task (i. e. type or duration of stimulus) on the training subset. The resulting selection of features is used as input for the SVM decoder (LIBSVM for Matlab [116]). Classification is performed using a one-vs.-one strategy.

In addition to decoding accuracies, results of the applied feature selection routine are used to compare the data with respect to the localization of channels that provide the most useful information. To do so, feature selection outcome—this means, which channels are selected—from all runs and folds of the CV routine is pooled to compute average selection rates (i. e. how often a channels is selected). These are illustrated in spatial distribution

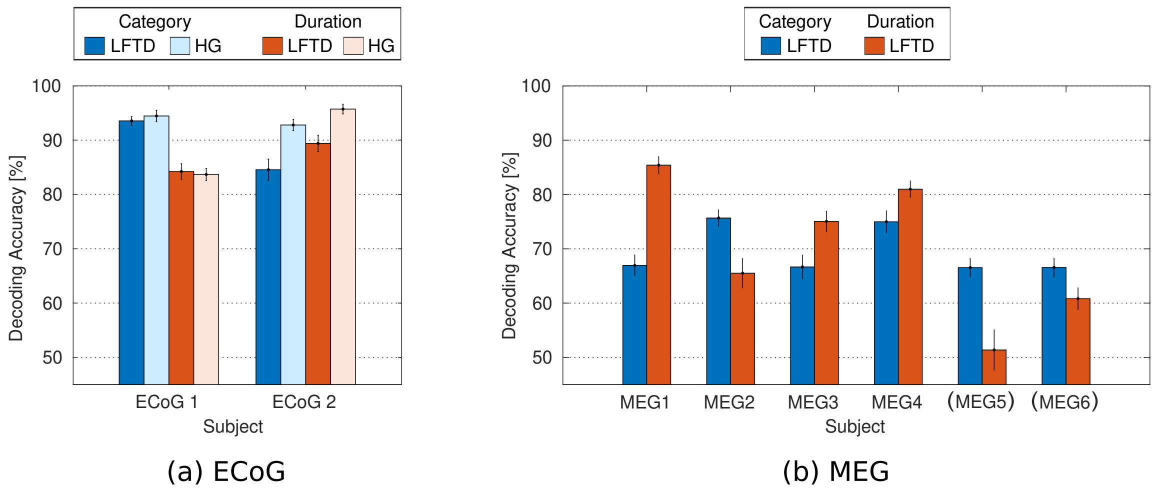


Figure 4.10: Decoding accuracies of category and duration decoding for all ECoG (a) and MEG (b) datasets. All results are averages across 50 repetitions of 5-fold CVs (error bars indicate standard deviation). Chance levels are 20 % for the duration task (five classes) and 33 % for category decoding (three classes). (Parts of this figure are reproduced from [120].)

maps, often called *channel maps*. For means of simplicity, interpretation of the channel maps with respect to involved brain areas is done on sensor level only for MEG data (i. e. no source analysis has been performed).

Results Figure 4.9 shows a comparison of LFTD features from two different channels of ECoG and MEG datasets. The plot displays the top two channels from feature selection on dataset ECoG 2 with respect to separability of different stimulus durations (task 2). Additionally, the top ranked channel from two MEG datasets (MEG1 and 4) are shown for comparison. Apparently, features extracted from ECoG have substantially higher signal-to-noise ratio. Still, qualitative similarities are found between ECoG and MEG features. In both cases, more or less prominent signal deflections mark stimulus onset (at the 0 ms mark) and offset. In case of ECoG signals, the appearance of additional stimuli within the trial segment of 2.1 seconds ('multi-stimulus-issue', cf. Section 3.7.2) can be seen clearly in form of distinct peaks that occur a certain time after stimulus offset—just like for the original stimulus at 0 ms. This effect is much less prominent in MEG, presumably because it is masked by the low signal-to-noise ratio (SNR). Different from the LFTD features, no usable HG features could be extracted from MEG recordings. As a consequence, these are left out in all subsequent analysis.

Decoding accuracies are assessed for both decoding problems using 50-times-5-fold CVs. For ECoG data, decoding is performed individually for both decoding problems and feature types, yielding a total of four different settings. Results for all datasets are shown in Figure 4.10. Decoding accuracies are quite similar for both of the ECoG subjects. Contrary, strong variation is found across the results for MEG datasets, especially for duration decoding. Results from subjects MEG5 and MEG6 are particularly low in the duration decoding task.

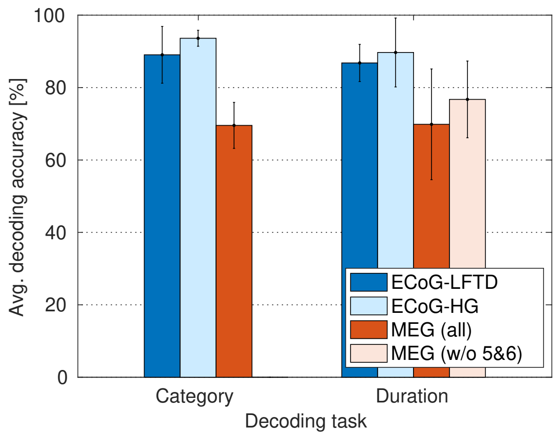


Figure 4.11: Average decoding accuracy in both decoding tasks for ECoG and MEG data. ECoG results are shown for LFTD and HG features. As no usable HG features have been found for MEG data, only the LFTD result is shown. Severe blink artifacts caused substantial performance issues for subjects MEG5 and MEG6 in duration decoding. Hence, a separate average is shown that does not take into account the affected datasets ('MEG (w/o 5&6)'). (Figure adapted from [120].)

Closer (visual) inspection shows that the data from these two subjects are heavily affected by blink artifacts (not shown here). These artifacts lead to improper feature selection and classifier training for duration decoding. The category decoding task is not impaired as severely.

Decoding accuracies have been averaged across subjects (Figure 4.11). With respect to the blink artifact issue with subjects MEG5 and MEG6 (see above), a separate average has been computed with the affected datasets left out ('MEG (w/o 5&6)'). Category decoding accuracy is significantly higher for ECoG data (both feature types) than for MEG data ($p^{\mathrm{LFTD}} = 0.0057$, $p^{\mathrm{HG}} = 0.0012$; difference t-test for $\mathrm{Acc}_{\mathrm{ECoG}} - \mathrm{Acc}_{\mathrm{MEG}} > 0$, $\alpha = 0.05$). For duration decoding, ECoG results are still higher than with MEG on average. However, differences between the results are smaller and have larger error bounds (non-significant difference for all combinations $p = 0.072 \ldots 0.145$). More detailed information on decoding outcome is available from the confusion matrices (CM, Figure 4.12). For the category task, ECoG and MEG results show the same characteristics. They differ almost exclusively in the absolute accuracy, which can be seen clearly when comparing the ECoG result (Figure 4.12 (a)) with the CM for MEG that has been rescaled to a min-max window (Figure 4.12 (c)). Decoding errors have nearly identical distribution. Face stimuli are decoded with the highest accuracy and only marginal confusion with the other classes. The discrimination between objects and watches seems to be more difficult for the decoder. Approximately 12 % (MEG: ~18 %) of these trials are mixed-up with the other class. As far as the duration task is concerned, CMs also appear qualitatively similar. The difference plot between both modalities (Figure 4.12 (f)) reveals that ECoG provides higher accuracy particularly for longer duration stimuli. Furthermore, subdiagonal CM entries - which refer to confusion between neighboring durations - are markedly smaller in the ECoG case.

Based on the selected channels in all runs and folds of the 50-times-5-fold CV routines,

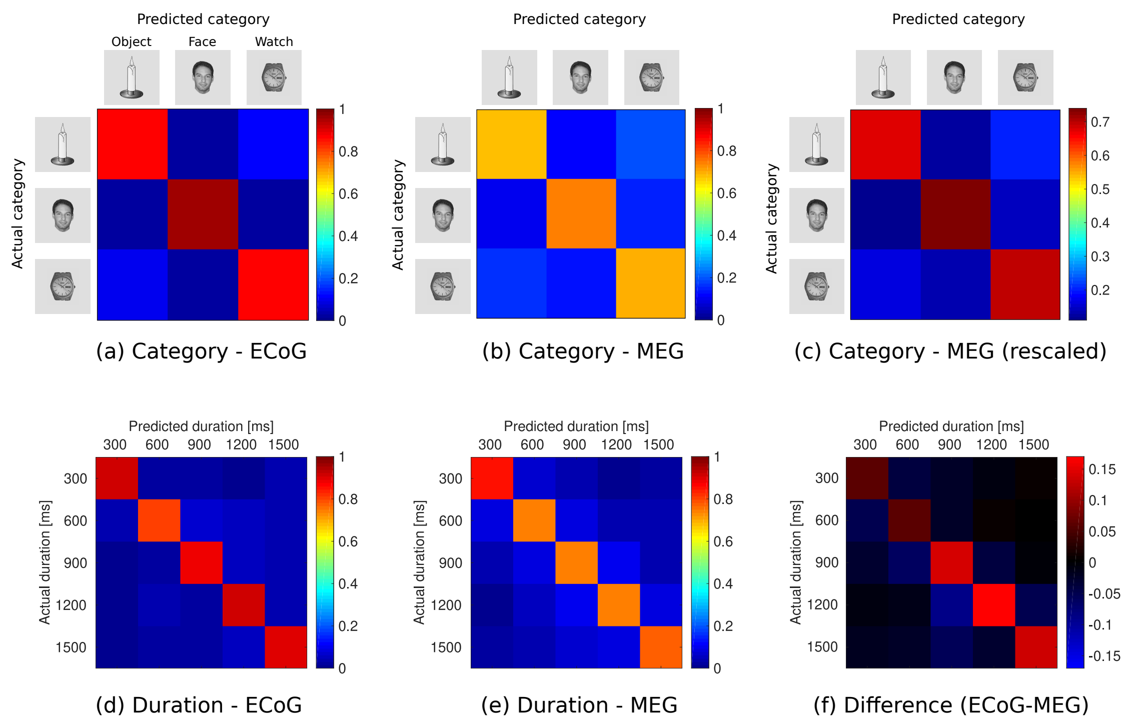
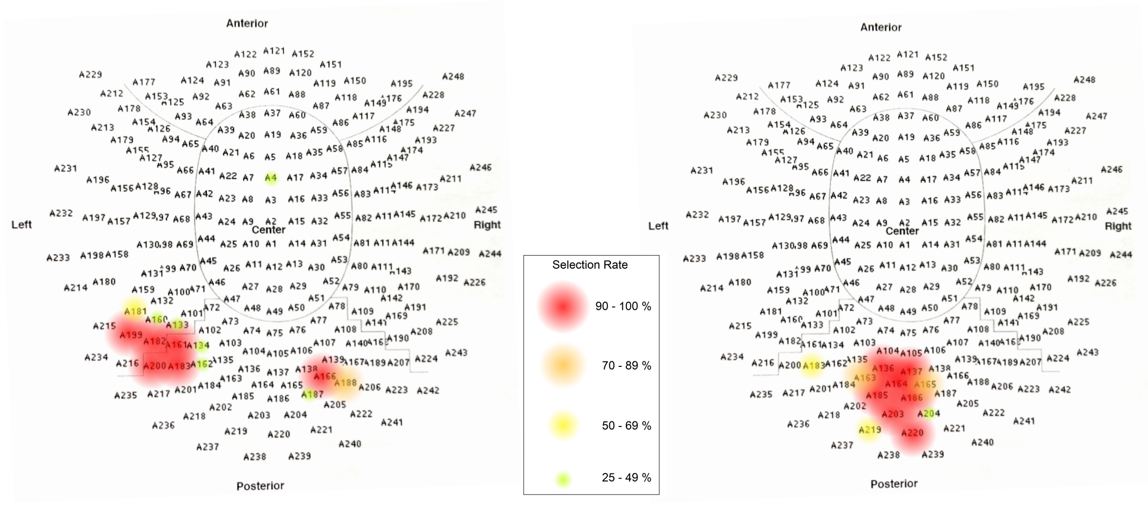


Figure 4.12: Confusion matrices (CM) of category (a-c) and duration decoding (d-f) results. The CMs are averages across all datasets from the respective acquisition modality. Sub-figures (a) and (b) show the results from category decoding with ECoG and MEG data, respectively. For sub-figure (c), the CM from (b) has been windowed with respect to the minimal and maximal value, to facilitate easier comparison to the ECoG results in (a). Sub-figures (d) and (e) depict the duration decoding results. To highlight the differences between ECoG and MEG results, sub-figure (f) shows the corresponding difference matrix.

channel maps have been computed. Selection rates are shown exemplary for subject MEG1 in Figure 4.13. For comparison, electrode grid locations including the dominantly selected electrodes from subject ECoG 2 are shown in Figure 4.14. In the duration decoding task, a strong tendency towards channels from the posterior region (channels A136-137, A164, A185-186, A203 and A220) can be seen in the MEG channel map. The location of these sensors implies that the corresponding brain signals originate from the occipital cortex (primary visual area, V1). Closely similar locations are found for selected electrodes in the ECoG recordings (cf. Figure 4.14 (b)). In the category task, primarily selected MEG channels are located more towards the sides, with a slight tendency towards the left hemisphere (sensors A161, A182-183 and A199-200; right hemisphere: A166). These sensors are likely to pick up signals from the temporal lobe. Dominantly selected electrodes in the ECoG datasets are located in the ventral temporal region. This indicates that the relevant information recorded with MEG comes from the same brain area. However, source analysis would be required to assess more precise information.



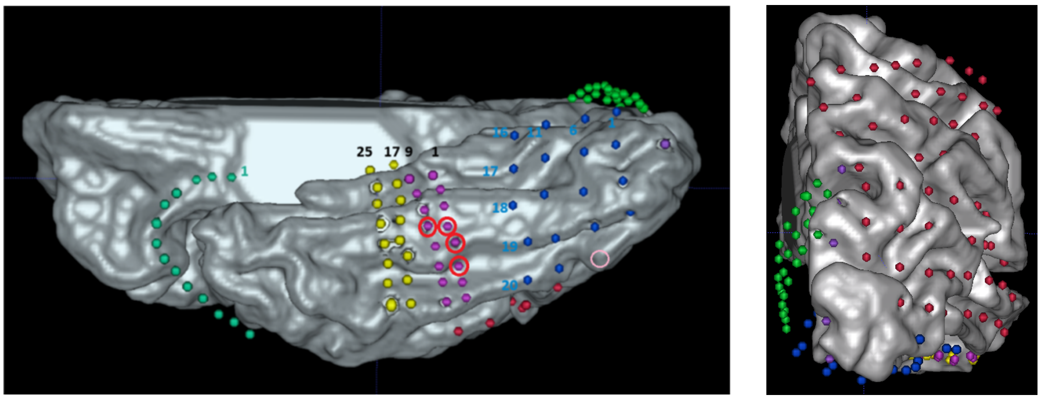
(a) Category

(b) Duration

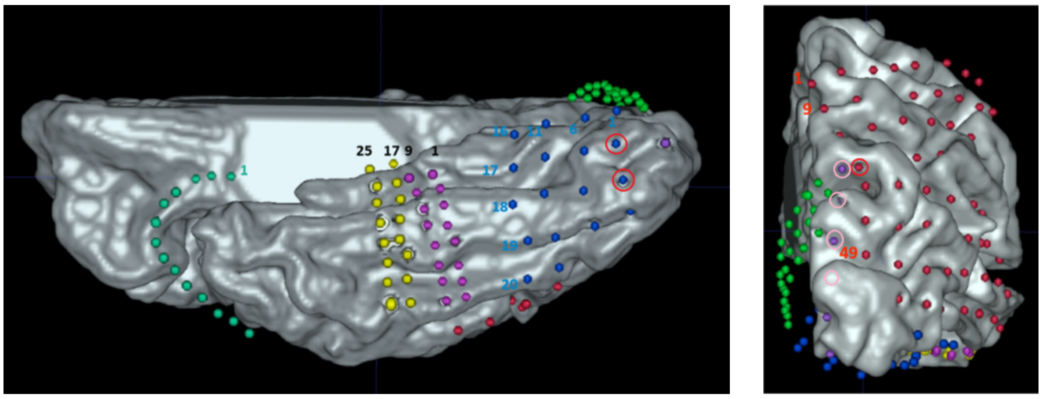
Figure 4.13: Spatial distribution of dominantly selected MEG channels in category (a) and duration decoding (b) from subject MEG1. Selection rates refer to the number of cases within the 50-times-5-fold CV that a certain channel has been selected. The maps show a two-dimensional mapping of MEG sensor location. Anterior channels correspond to frontal brain areas, whereas posterior channels pick up signals from the occipital cortex.

Discussion The results of this section indicate that MEG recordings have qualitatively similar properties as ECoG data in the picture category experiment. Extracted LFTD features show similarities with respect to signal deflections associated with picture onset and offset. Unfortunately, MEG data seems to contain no usable high frequency information, which is in accordance with previous observations in other MEG data from the same acquisition setup [121]. This is most likely due to the low amplitude of these signals and the rather low SNR of the MEG signals. Consequently, no meaningful HG features could be extracted from the MEG recordings. However, the achievable decoding accuracies for ECoG show that LFTD features provide almost the same level of accuracy as HG features in both investigated tasks. Hence, being able to only use LFTD in MEG recordings does not severely limit its usefulness. Overall, decoding accuracies are higher with ECoG recordings (significant[4] for category), which is expected due to the higher signal quality of the invasive recordings. In the duration task, two of the datasets provide particularly low accuracy. This has been identified to be caused by severe blink artifacts. These create signals with extremely high amplitudes, misleading the feature extraction routine to select the affected channels. For actual decoding, these channels do not provide information that are consistent enough. To avoid any bias, these two datasets have been left out in further analysis. Detailed information on individual decoding errors from the CMs (Figure 4.12) is of particular interest to study the comparability of both modalities. In the duration

[4]It should be noted that significance analysis has very limited power in this study due to the extremely low number of datasets (2 and 6).



(a) Category



(b) Duration

Figure 4.14: Anatomical electrode grid location of subject ECoG 2. The pictures show a ventral (left column) and posterior (right column) view of the right hemisphere. The patient has been implanted with a variety of electrode grids and strip electrodes, covering large areas of the occipital cortex as well as ventral temporal areas. Dominantly selected electrodes for category (a) and duration decoding (b) are highlighted with red and pink (less dominant) circles. Images of electrode reconstructions courtesy of Helen Wills Neuroscience Institute, UC Berkeley.

task, superiority of ECoG is found. Particularly for neighboring durations, a non-negligible amount of mix-ups occurs in MEG decoding. As a consequence, it has been decided to change the paradigm as far as the different presentation durations are concerned. Instead of five different durations, in the altered paradigm this count is reduced to three classes that are more spread out (see Section 3.7.2, 'MEG – modified paradigm'). The modified paradigm is used to re-record MEG data for database extension for the next section.

Comparison of the CMs for the category task shows that both modalities behave closely similar. In both cases, face trials are predicted correctly with high accuracy and almost no confusion with the other stimuli types. Mix-ups between the other two categories have the same ratio (w. r. t. the overall accuracy) in both modalities. In addition to the similarities in decoding error distribution, dominantly selected channels have been identified to roughly correspond to the same areas of the cortex. Identified category-selective regions (mainly ventral temporal regions) are in accordance with the literature (e. g. [122]). As mentioned before, these statements have entirely qualitative character for the MEG datasets, which is sufficient for the demands of this investigation. For applications that require accurate quantitative comparison (e. g. grid implantation planning), source analysis would be required to allow for precise anatomical localization of signal origins.

Conclusion In summary, the findings strongly indicate that MEG recordings provide the required signals from the brain regions of interest with sufficient quality for visual tasks. This shall be used to add additional datasets for analysis in the next section, which will deal with detailed analysis of the picture category task with HMM decoders and will focus primarily on category decoding. In that particular decoding task, two of the MEG datasets have shown markedly higher decoding accuracies (75 % vs. 66 %). To provide the best possible comparability for analysis, these two subjects are chosen to extend the database. As mentioned before, the paradigm has been slightly modified for technical reasons (MSI, see Section 3.7.2, 'MEG - modified paradigm') and to adjust the level of difficulty for fairer comparison. Data has been re-recorded from the two mentioned subjects (MEG2 and MEG4) with the modified paradigm for the following section.

4.3. Extraction of additional information: *Duration decoding*

The findings from the first study (Section 4.1) have shown that HMMs can reach the performance of the gold-standard static classifier SVM. In this section, investigations focus on the use of beneficial properties of the dynamic nature of HMMs. By concept, dynamic classifiers model the time sequence of the input signal. This circumstance shall be used to decode additional information from the data without a dedicated training step. As there is no direct analogue for this using static classifiers, it demonstrates the superiority of dynamic approaches in complex decoding tasks.

The central findings of this section have been published in [114].

Data Results presented in this section are based on data from the picture category task (Section 3.7.2). ECoG data from two patients is available for this study. As mentioned in the previous section, MEG data is used to extend the database for analysis. The investigations from Section 4.2 suggest that data from two of the investigated subjects are best suited for this purpose. A slightly modified paradigm (Section 3.7.2, 'MEG – modified paradigm') is used for MEG recordings to better meet the requirements for the analysis. In combination with the ECoG recordings, a total of four datasets constitutes the database. Details on how data are epoched into trials as well as information on pre-processing steps can be found in Section 3.7.2. As usual, data were inspected visually for bad channels and artifacts. Affected trials are removed from analysis (full breakdown in Table 4.6).

Approach Feature extraction is performed with the same settings as in the previous section. Low frequency time domain (LFTD) features are computed according to the method described in Section 3.1.1 with a cut-off frequency of 30 Hz for ECoG data and 10 Hz for MEG. To reduce feature dimensions, the resulting time series are down-sampled to 98 sample points (corresponding re-sampling rate: ~48 Hz). High gamma (HG) features are extracted from the frequency band of 70–200 Hz with a window length of 250 ms. Feature

Table 4.6: Information on count of channels and full breakdown of recorded and rejected trials for all subjects.

Subject	Acquisition modality	Number of channels	Recorded trials	Trial breakdown (objects/faces/watches)	
				rejected	remaining
ECoG 1	ECoG	101 electrodes	312	6/4/3	98/100/101
ECoG 2		110 electrodes		18/19/19	86/85/85
MEG 1 (MEG2)*	MEG	248 sensors	225	6/1/11	69/74/64
MEG 2 (MEG4)*				10/5/6	65/70/69

* Subjects MEG2 and MEG4 from Section 4.2 have been renamed to MEG 1 and MEG 2, respectively.

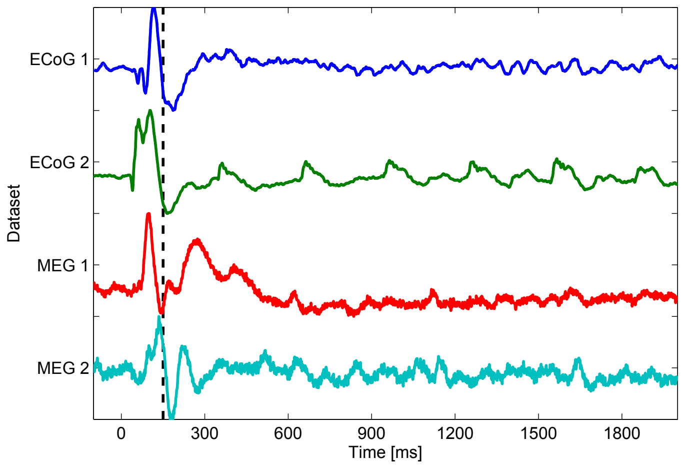
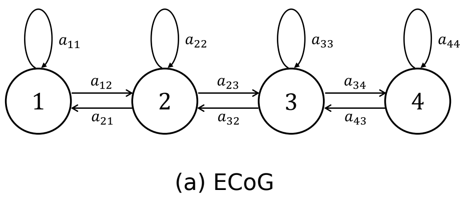


Figure 4.15: Examples of visually responsive channels for all four subjects. The graphs show raw signals averaged across trials of all categories and durations. Data have been normalized (i. e. scaled to the interval [0,1]) for better comparability. The dashed black line indicates the 150 ms time point. Channels with the highest average signal amplitude up until this time point are selected for decoding (up to a total of seven 'duration channels'). (Figure reproduced from [114].)

values are computed for 100 equidistant time points. This corresponds to an overlap between adjacent windows of about 92.5 %. As a consequence, HG features vary rather slowly over time (cf. Figure 4.18). Investigations from the previous section showed that no usable HG can be extracted from the available MEG data. For that reason, analysis of the MEG datasets is limited to LFTD features.

With respect to the objective of decoding both category and duration information from the data, a two-stage approach for channel selection is applied.[5] In a first step, channels are identified that carry relevant information for the category decoding task. This is done using the routine based on the DB-index (Section 3.2) that has also been used in the previous sections. Additionally, a second set of channels is sought-for that brings in information on the stimulus duration. To keep the overall strategy of predicting the duration training-free, an unsupervised approach is used to determine suitable 'duration channels' as follows: Raw signals of all trials are averaged separately for each channel; this results in a mean signal amplitude time course for each of the channels. Channels for which this mean amplitude exceeds a defined threshold value within the first 250 ms of the time series (i. e. up to 150 ms after picture onset) are selected for the set of duration channels. This specific time frame is considered, as it turns out to match the typical interval that contains the brain response to the appearance of a visual stimulus for both types of data (Figure 4.15). Hence, if the mean amplitude of a certain channel exceeds the threshold within that interval, this indicates that it represents neural ensembles that take part in the processing of early visual information. In case the number of channels fulfilling the threshold condition is higher (or lower) than the requested number, the threshold value is increased (or decreased) until the

[5]Section 4.2 showed evidence that information on category and duration are encoded in different brain areas (cf. Figures 4.13 and 4.14). Hence, separate selection of suitable channels is indicated.



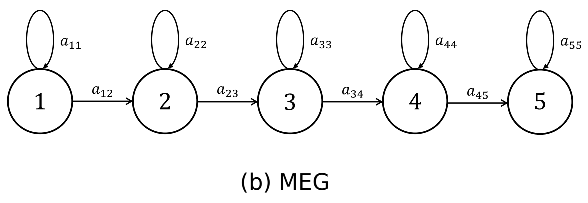


Figure 4.16: HMM topology used for (a) ECoG decoding and (b) MEG decoding. Besides state looping, the ECoG model allows transitions from a state to its predecessor and successor. In the MEG model, only single-stepped forward jumps are allowed (in addition to loops). The prior is set to force models to start with the first state in both cases. (Figure reproduced from [114], rearranged.)

target count is reached. It should be noted again that this routine is independent from any labels. Motivated by previous findings (Section 4.1, Table 4.3), a total number of ten channels is selected. The actual ratio between category and duration channels constitutes a trade-off between optimal category decoding performance and a sufficiently substantial incorporation of information on the stimulus duration. As a compromise, three channels are dedicated to category decoding. This is loosely motivated by the fact that there are three category classes.[6] The remaining seven channel slots are filled with 'duration channels'.

Due to fundamental structural differences between the two feature types, specific model topologies are used for each purpose (Figure 4.16, detailed discussion later). Generally, topologies are oriented towards findings from Section 4.1. The model used for MEG data (LFTD features) consists of five states, each containing a single mixture component. Findings from Section 4.1 further suggest that strict constraints are of high importance for time domain features (cf. Figure 4.8). Therefore, the MEG model allows only single-stepped forward jumps (in addition to state looping). Contrary, for the ECoG model (HG features[7]) a single-step backward jump is introduced. This was found to compensate for the multi-stimulus issue (see discussion). The backward jump allows the model to re-use earlier states for later segments within the time series. As a consequence, the total amount of states has been reduced to four. The different feature properties also imply specific requirements for model initialization. The time discriminative clustering routine (Section 3.3.3) takes into account the temporal structure of the input data by introducing the time point of a feature sample as an additional feature dimension. LFTD features display a rather complex temporal structure compared to HG features. Consequently, different values have been used for the time coupling constant τ in the clustering routine: $\tau_{\text{LFTD}} = 25$ and $\tau_{\text{HG}} = 0$.

Individual HMMs are trained for each of the category classes. This results in a total of three HMMs (objects, faces, and watches). HMM functionality is provided by the HMM toolbox for Matlab from Kevin Murphy [115]. The Baum-Welch algorithm is used for estimation of all model parameters (i. e. state means, covariance matrices, and state transition probabilities). Classification of picture category is performed with a straightforward maximum likelihood approach; the model with the highest probability determines the assigned

[6]Note that this is an entirely empirical choice that cannot be justified theoretically.

[7]As it turns out, LFTD features do not allow for meaningful duration analysis in case of ECoG data. This aspect is elaborated on in the results (Figure 4.19) and discussion paragraph. Consequently, results for the ECoG datasets are presented for HG features only.

label. The full procedure is illustrated in Figure 4.17.

After category classification, the Viterbi path V_t of the "winning HMM" is calculated. This is then used to extract information on the duration of the stimulus as follows: First, a threshold value θ needs to be defined. The choice of θ will be discussed later. Based on the chosen threshold, the Viterbi paths are scanned for the first sample point s that reaches the threshold:

$$s = \min_t t, \quad \text{s. t.}\, V_t \geq \theta. \tag{4.1}$$

Using a linear relation

$$d = m \cdot s + n, \tag{4.2}$$

the stimulus duration d is predicted from the sampling point s. Equation (4.2) requires a slope m and the offset n. The slope is known *a priori*, as is corresponds to the time increment from one feature sample to the following. If τ denotes the length of a trial in seconds[8] and S is the number of samples in the extracted feature vector, the slope is given by $m = \tau/(S-1)$. The offset n is introduced to compensate for eventual delays between the end of stimulus presentation and the according state changes of the corresponding HMM. Additionally, the offset can also account for delays between event and brain response. Unlike the slope, the offset n is not simply an inherent value of the signal processing chain. To determine an appropriate value for it, a calibration step is necessary. This is done using the (duration) labels of a single duration class *cal* (e. g. all 300 ms trials). For these trials, the threshold crossing samples s_i are computed using Eq. (4.1). From all these points, a representative (mean) sampling point $s_{\text{rep}}^{(cal)}$ is calculated with a histogram-based method. First, a rough histogram is computed by dividing the data into ten equally sized sample boxes b_j. From these, the box with the highest count is selected ($b_{\max}$). In the second step, the median of all samples in the proximity (± 10 sample points) of the center c of the selected box $b_{\max}$ is calculated:

$$s_{\text{rep}}^{(cal)} = \underset{s_i \in [c-10, c+10]}{\text{median}} (s_i) \,.$$

Using the duration $d^{(cal)}$ of the class that has been used for calibration (e. g. 300 ms), Eq. (4.2) can be used to estimate the offset parameter $n^{(cal)}$. Ideally, the offset parameter would be equal, irrespective of the calibration data. However, in practice, the result does depend on the actual choice of the calibration class *cal*. Consequently, the presented routine is applied once for each of the duration classes—almost in a cross-validation-like manner—to provide objective results.

$$n^{(cal)} = d^{(cal)} - m \cdot s_{\text{rep}}^{(cal)}, \;\; cal = \begin{cases} 1, ..., 5 & \text{ECoG} \\ 1, 2, 3 & \text{MEG} \end{cases} \tag{4.3}$$

It is important to note that this strategy cannot be interpreted as a classifier training, as labels from only a single class are made available to the decoder. Moreover, postulating

[8]Note that for HG features, the window length l_w (in seconds) of the power spectrum computation must be taken into account. Hence, τ needs to be reduced by that length: $\tau^{\text{HG}} = \tau - l_w$.

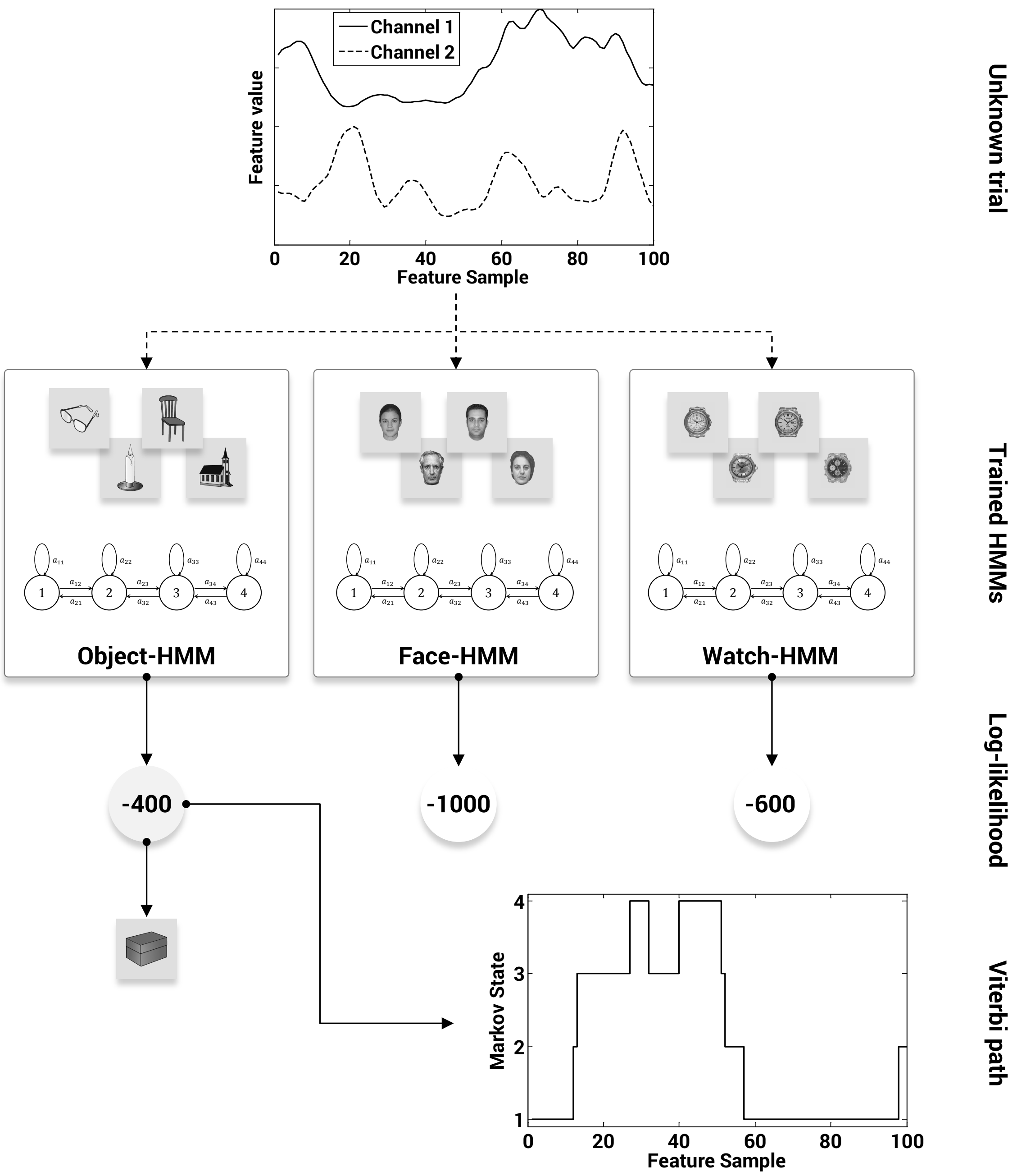


Figure 4.17: Schematic illustration of the decoding routine. The trained HMMs are used to model the feature sequence that has been extracted from the unknown trial. Each of the models provides a likelihood for generating that particular feature sequence. Category classification is decided by means of a straightforward maximum-likelihood approach, i. e. the model with the highest (log-)likelihood determines the assigned label. The Viterbi path (i. e. state sequence that is most likely to explain the given feature sequence) of this model is extracted and used for duration decoding. (Figure reproduced from [114].)

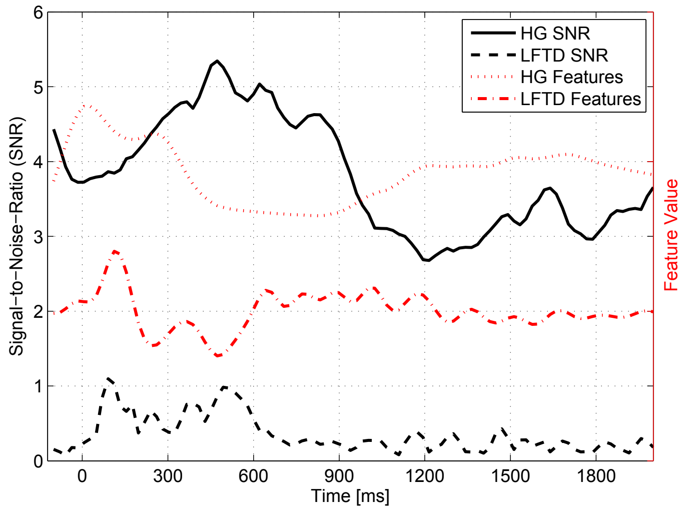


Figure 4.18: Average feature values (red) and corresponding signal-to-noise ratios (SNR, black) for HG and LFTD features. Mean μ and standard deviation σ across all trials with a presentation duration of 300 ms from subject ECoG 2 are computed for all ten selected channels. SNR is computed for each individual channel as the quotient of $|\mu| / \sigma$ and averaged afterwards. Brain response to picture offset ends around 600 ms. At this point, SNR of LFTD (dashed black line) nearly drops to noise level and consequently appropriate modeling of these time segments becomes nearly impossible. Contrary, HG features show high SNR (solid black line) throughout the entire duration. Additionally, the period of increased SNR extends over a longer duration than for LFTD (approximately until 900 ms). (Figure reproduced from [114].)

the offset as *a priori* knowledge—for example based on findings from other subjects—could also be a conceivable solution. Such a routine would allow for fully unsupervised decoding of duration information. This aspect will be discussed later.

Results As mentioned before (cf. Sections 3.7.2 and 4.2), the paradigm used for ECoG subjects contains the multi-stimulus-issue (MSI), i. e. additional stimuli can appear within the time series of trials with briefer presentation durations. The time domain features (LFTD) have a quite complex structure along with rather low SNR (Figure 4.18). In combination with the limited amount of training data, this poses substantial difficulties for the HMM decoder, which is therefore not capable of distinguishing the differences between actual stimulus offsets and 'ghost' events resulting from the MSI. To prove that hypothesis, a data subset has been created, in which the MSI is eliminated by narrowing the investigated time interval to [-100, 1200] ms (and removing the few remaining MSI trials[9]). The resulting dataset allows to compare model outputs of a decoder that is trained on MSI-spoiled data (full dataset) or on an MSI-free training set. Figure 4.19 shows that the decoder trained on full length data outputs results with indistinguishable Viterbi paths. Contrary, the model

[9]These can still occur for combinations with 300 ms stimulus duration and 600 ms inter-stimulus-interval. However, their count is low enough to simple exclude them from the dataset without losing too much data.

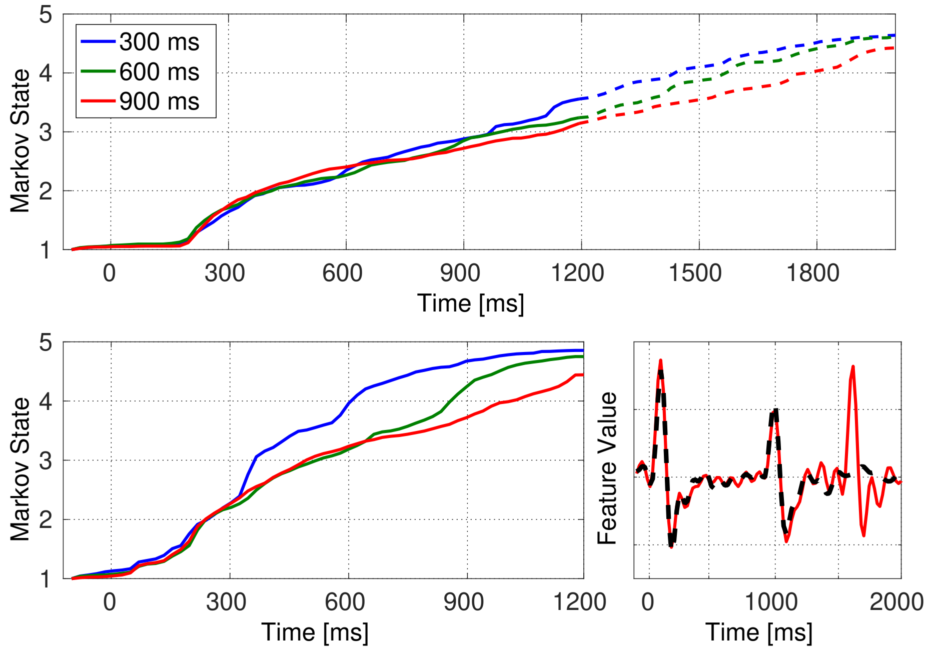


Figure 4.19: Comparison of mean Viterbi paths with and without trials containing multiple stimuli (multi-stimulus-issue, MSI). The graph on the top shows Viterbi paths for full length ECoG trials (subject ECoG 2) using LFTD features. These trials can contain additional stimuli in the second half of the segment (cf. Section 3.7.2, Figure 3.5). Viterbi paths have been averaged separately for trials of different duration. It is clearly visible that the paths of all durations are almost identical in the relevant time interval (solid lines), which makes duration decoding impossible. To assess whether this is caused by the MSI, decoding has also been performed with a decoder that is trained on shortened versions of the trials. In this setting, trials are truncated at 1200 ms and are free of additional stimuli. Corresponding Viterbi paths (bottom left) show clear differences between stimuli of different duration. For comparison, mean feature values are illustrated in the bottom right plot. The graph shows LFTD features from a typical 'duration channel' for trials with a stimulus duration of 900 ms. Averaging has been performed separately for trials that do not contain additional stimuli (dashed black line) and for those trials in which an additional stimulus occurs at 1500 ms (solid red line). (Figure reproduced from [114].)

that has been trained on the MSI-free data produces Viterbi paths with the expected characteristics.[10] As a consequence, LFTD features are not considered in the analysis of the ECoG datasets. Instead, full focus is laid upon HG features, which do not suffer from this problem (see discussion).

In the category decoding task (three-class problem), decoding accuracies up to 80 % can be achieved with the HMM decoder for the ECoG datasets (Figure 4.20). For means of comparison, the same decoding has also been performed using gold-standard SVM classifiers (LIBSVM for Matlab [116], one-vs.-one). The highest accuracy reached with SVM is 85 %. For MEG data, decoding results are worse for both classifiers. HMMs provide about 20 % and SVMs about 15 % less accuracy (absolute differences). Hence, the performance gap

[10]Note that these aspects might be significantly easier to understand after going through the main results of this section.

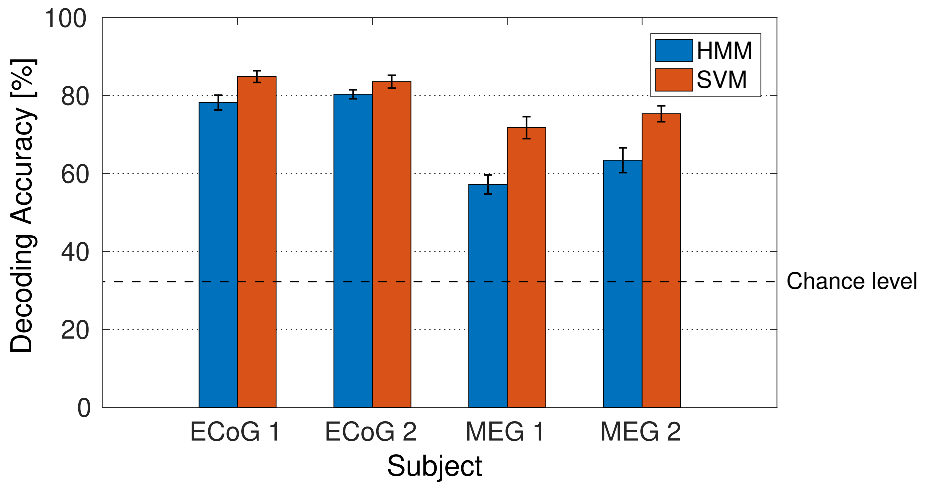


Figure 4.20: Category decoding accuracies of HMM and SVM decoding for all four datasets. The shown results are average performances from 20-times-5-fold CV procedures (error bars indicate standard deviations). All performance differences are statistically significant (two-sample one-sided t-test, $\alpha = 0.001$). Chance level for this three class problem is $33.\bar{3}$ %. Tabulated results are provided in the Appendix, Table A.5. (Figure reproduced from [114], recolored.)

between SVM and HMM is about 12–15 % for MEG data, whereas for ECoG, HMMs nearly reach the SVM performance (about 3–6 % difference). This correlates well with findings from Section 4.1 showing that differences in the accuracy of both decoders get smaller for higher overall accuracy (resulting from higher quality data). More detailed information of decoding performances can be taken from the confusion matrices (Figure 4.21). These show consistent behavior across all datasets. By far, faces are detected best, whereas there are frequent mix-ups between object and watch trials. This holds true for ECoG as well as MEG data.

To assess duration information, Viterbi paths have been analyzed according to the routine described in paragraph 'Approach'. The extracted Viterbi paths for all single trials are illustrated in Figure 4.22. Apparently, the paths of trials with the same stimulus duration show similar behavior. The offset of stimulus presentation is indicated by the red triangle in Figure 4.22 to facilitate interpretation. It becomes apparent that there is a consistent state change across single trials that is (temporally) correlated with the end of stimulus presentation. This appears like a flank in the plot in Figure 4.22.

The mean over all Viterbi paths belonging to stimuli of a certain duration has been computed (Figure 4.23). It can be clearly seen that the average paths of different presentation durations are distinguishable. In particular, a tendency is observed that the time points at which the paths start to deviate from each other correlate with stimulus presentation duration. In the following, it will be assessed whether these observations also hold true on a single-trial level, and hence, can be used to decode duration information.

As described in the 'Approach' paragraph, calibration is required to determine the threshold parameter for duration estimation. To do so, every duration class has been used once

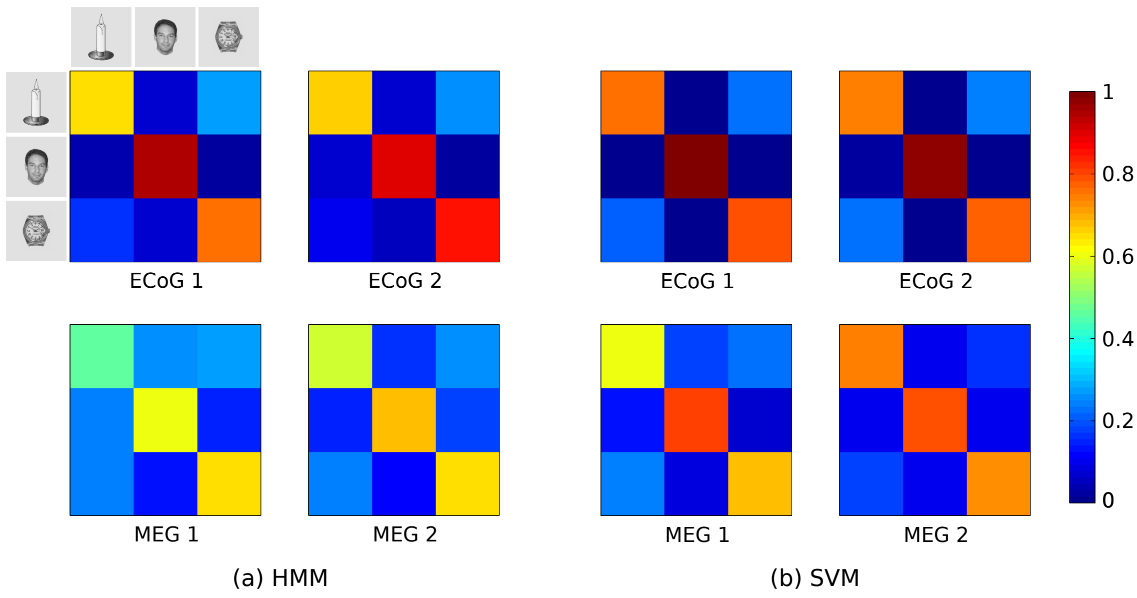


Figure 4.21: Confusion matrices of the category decoding task for all four data sets. (a) HMM decoder, (b) SVM decoder. For an explanation of confusion matrices, refer to the caption of Figure 4.6. (Figure reproduced from [114], re-arranged.)

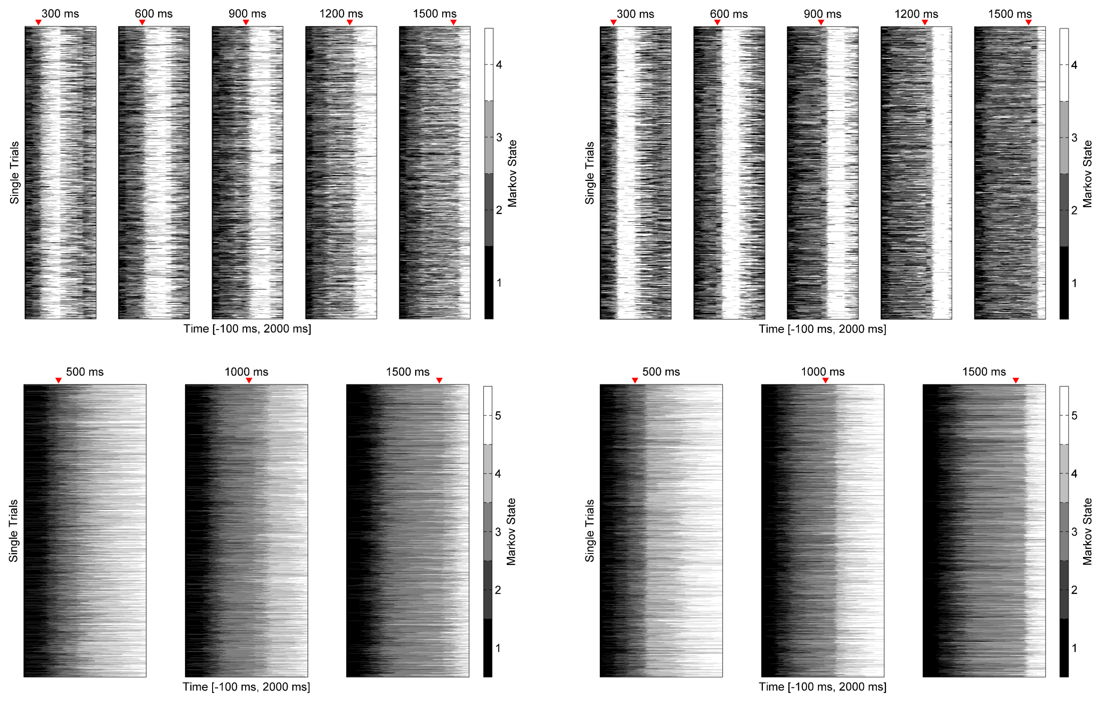


Figure 4.22: Single trial Viterbi paths for all datasets (top row: subjects ECoG 1 and 2, bottom row: MEG 1 and 2). Trials are grouped with respect to the presentation durations. The red triangles indicate the end of stimulus presentation. (Figure reproduced from [114].)

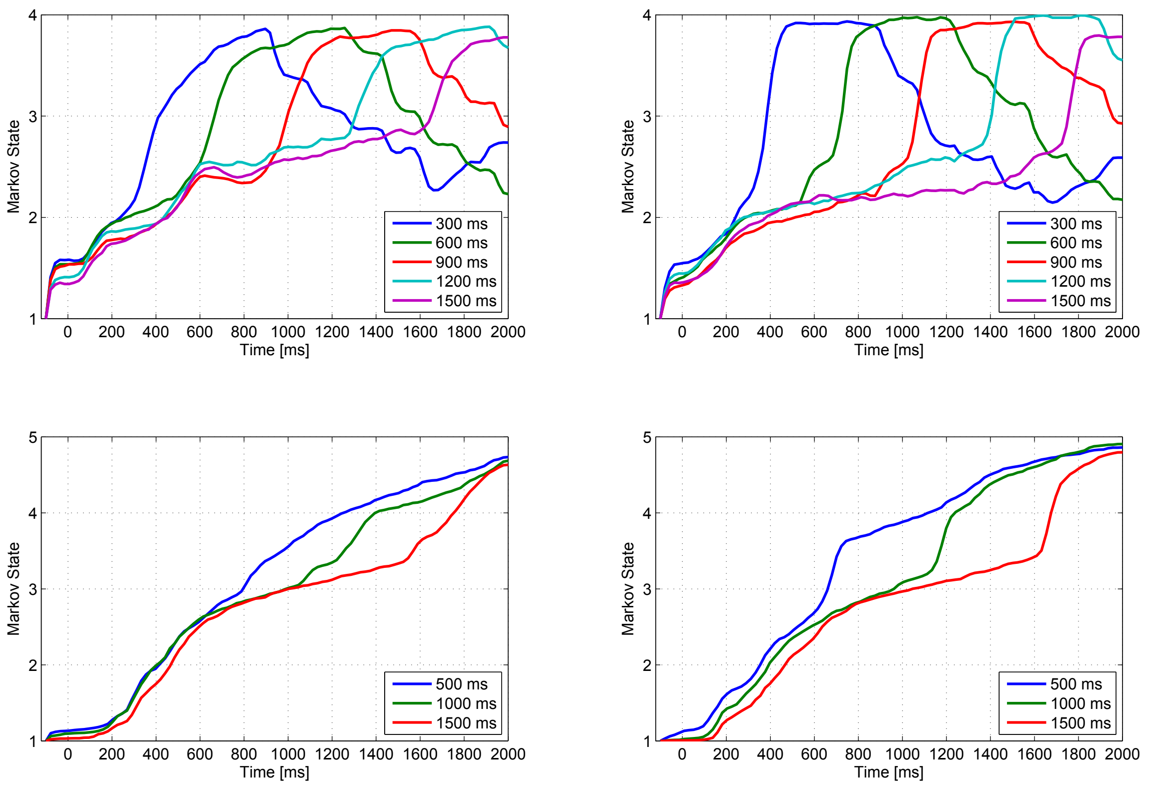


Figure 4.23: Mean Viterbi paths for all datasets (top row: subjects ECoG 1 and 2, bottom row: MEG 1 and 2). Single trial Viterbi paths have been averaged individually for the different presentation durations (irrespective of category). (Figure reproduced from [114].)

for computation of the threshold (cf. Eq. (4.3)). The determined value is then used to predict stimulus duration for all remaining trials using Eq. (4.2). Results for all calibration sets (five/three in ECoG/MEG, respectively) are documented in Table 4.7. Accuracy is evaluated by means of root mean square errors (RMSE) of predicted durations compared to actual durations. A final performance measure is assigned by averaging the results from all calibration cases (individually for each dataset).

For comparison, chance values are quantified by performing the same evaluation routine but instead of predicting durations using relation (4.2), random durations from the interval [0, 2000] ms have been assigned to each trial. To simplify interpretation of the results, all predicted durations have been mapped to the appearing discrete classes (i. e. 300 ms, 600 ms, etc. for ECoG and 500 ms, 1000 ms, 1500 ms for MEG) by means of simple assignment to the closest value. This allows for computation of conventional decoding accuracies that document how many predictions have been correct. Additionally, confusion matrices can be analyzed to shed light on the error distribution; these are shown in Figure 4.25. Frequent mix-ups are found for the shortest durations, i. e. between 300 ms and 600 ms in ECoG and between 500 ms and 1000 ms for MEG. Except for that, decoding accuracy tends to decrease for longer presentation durations.

Table 4.7: Estimated offset parameter $n^{(cal)}$ in ms for all calibration sets and subjects.

	Calibration set *cal*					
Subject	1	2	3	4	5	Mean
ECoG 1	-148	-110	-90	-89	-107	-108.8
ECoG 2	-148	-147	-165	-164	-163	-157.4
MEG 1	-421	-350	-236			-335.7
MEG 2	-293	-286	-257			-278.7

Table 4.8: Number of states Q used in the HMMs, threshold state θ for duration decoding and corresponding accuracies (chance levels in brackets) for all data sets. For comparison, accuracies of supervised decoding (HMM classifier) of duration classes are shown in the last column.

			Duration		
Subject	Q	θ	RMSE (ms)	discrete (%)	supervised (%)
ECoG 1	4	4	279 (600)	53.3 (20.0)	30.8
ECoG 2	4	4	123 (600)	82.8 (20.0)	37.2
MEG 1	5	4	232 (585)	68.5 (33.3)	65.9
MEG 2	5	4	187 (585)	76.5 (33.3)	71.9

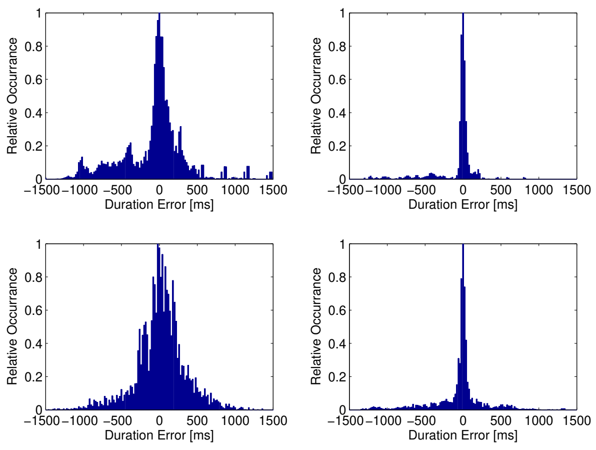


Figure 4.24: Histograms of duration decoding errors for all datasets (top row: subjects ECoG 1 and 2, bottom row: MEG 1 and 2). For means of better comparability, relative occurrence rates are shown instead of actual counts (i. e. histograms are normalized by the highest appearing value). This allows to better inspect the shape of the histograms. Negative error values indicate that the predicted duration is shorter than the actual. (Figure reproduced from [114].)

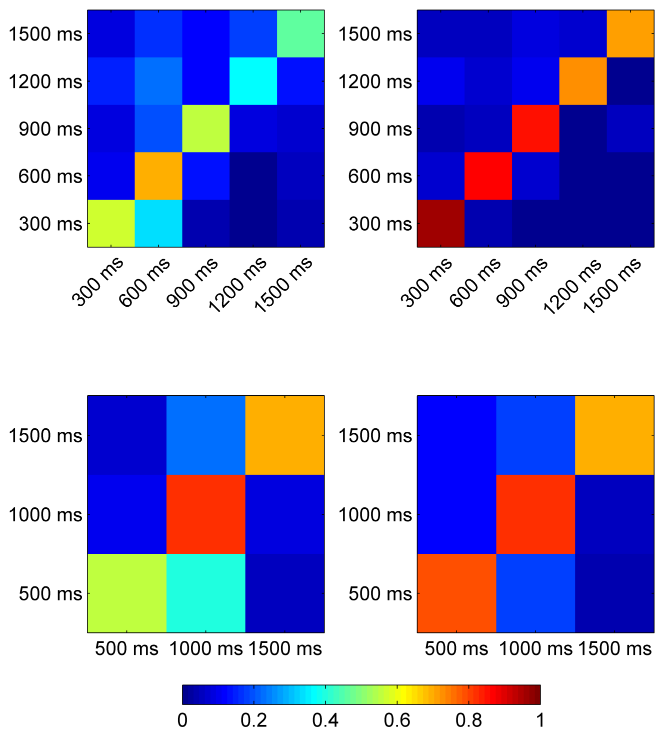


Figure 4.25: Confusion matrices for duration decoding (top row: subjects ECoG 1 and 2, bottom row: MEG 1 and 2). (Figure reproduced from [114].)

Discussion All results show consistent behavior for both acquisition modalities and all datasets. In category decoding, frequent mix-ups appear between watches and objects. This is expected due to the close similarity of these image types; in fact, watches are usually considered to be a sub-category of objects. From the three occurring picture categories, faces are decoded with the highest accuracy. This is in accordance with other studies reporting similar findings [123–125]. With respect to the overall decoding accuracy, the SVM decoder provides superior results for MEG data. Since feature quality is lower for MEG datasets, this observation is consistent with the findings from Section 4.1. For ECoG data, which provide higher signal quality, the performance gap is markedly smaller.

As mentioned in the ’Approach’ paragraph, the differences in the properties of the two investigated feature types (i. e. LFTD and HG) made it necessary to adapt the model structure and decoding strategy accordingly. In case of LFTD features, task-related information is masked by the ongoing brain background activity. This is superimposed to the signal with amplitudes in similar ranges as the actual response to the image on- and offset. Therefore, time segments succeeding the offset of a picture (i. e. during inter-stimulus-intervals) do not contain reproducible LFTD feature sequences. As a consequence, these segments can only be modeled appropriately using very simple HMM topologies. This is addressed here by using a pure left-to-right model that restricts model transitions to looping and single-stepped forward jumps. HG features have different characteristics, which allow for a more complex approach. The overall SNR for HG features (Figure 4.18) is substantially higher, and additionally, SNR does not degrade as much as for LFTD in periods of picture absence. This circumstance ensures that consistent information on these episodes is available to the model and hence, this can be utilized in model structure. Consequently, HMMs used for HG decoding are defined with a topology that also allows for a single-stepped backward jump in addition to the left-to-right structure for LFTD features. Due to the backward jump, earlier states are reused for modeling of a given time sequence. As a result, less states are required to appropriately represent the incoming signals. The differences in model topology are clearly reflected by the corresponding Viterbi paths. Since Viterbi paths represent the state sequence that is most likely to reproduce the given feature sequence, they indicate how the HMM models the signal. For LFTD features (MEG datasets), mean paths typically show the following structure (cf. Figure 4.26 (a)):

I Dwell in initial state for approx. 200 ms.

II Steady increase up to approximately state three.

III Sharp bend, followed by segment with steep slope.

IV Segment with slight slope, ascent to highest state.

The first segment can be interpreted as the pre-stimulus period (i. e. time interval before picture onset). Segment II has variable length—depending on the stimulus duration—with longer duration stimuli leading to a longer segment II. The paths continue with segment III after a sharp bend. Typically, the Viterbi path crosses the threshold state θ within this segment. Hence, segment III can be associated with the picture offset.

The model used for ECoG datasets results in Viterbi paths with different properties (cf. Figure 4.26 (b)):

I Early jump to state 2.

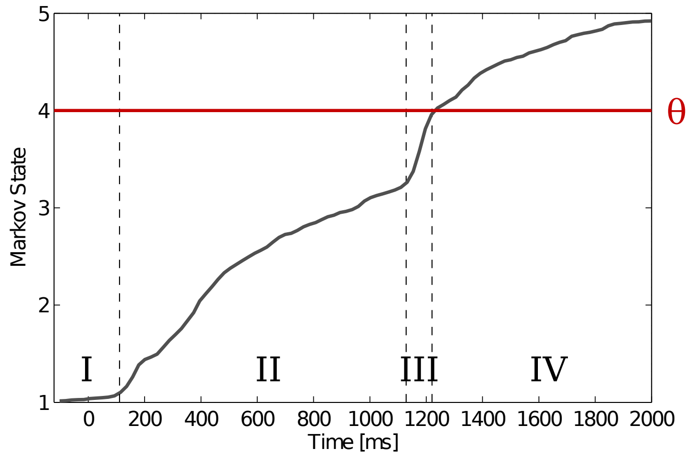
II Segment with very shallow slope.

III Sharp bend, followed by segment with steep slope.

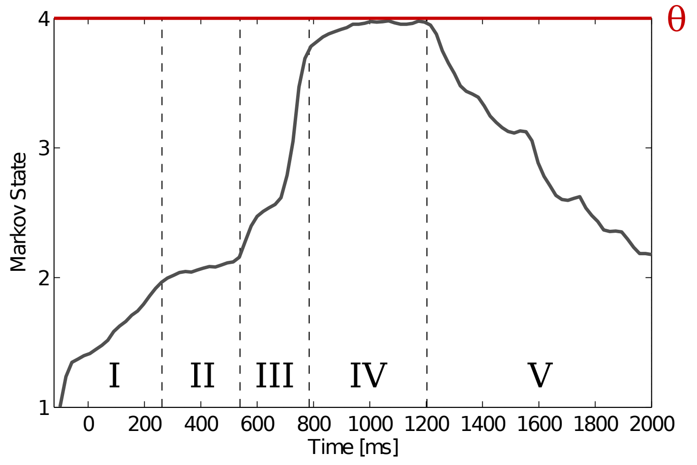
IV Marked plateau, dwell in highest state.

V Descent to earlier states (approx. to state 2).

The early jump to the second state in segment I is present in paths of all stimulus durations. Hence, it can be associated with the picture onset. Segments II and III have similar properties as their counterparts from the LFTD setting. For longer duration stimuli, segment II extends over a longer period. As in the LFTD case, segment III consists of a steep-sloped part; at its end, most paths reach the highest state (in this case: state 4). Note that Figure 4.26 depicts mean paths (averaged across all trials of a certain duration) and hence, the shown curve does not reach state 4 at this point, although the majority of



(a) LFTD (MEG)



(b) HG (ECoG)

Figure 4.26: Structure of mean Viterbi paths for MEG (a) and ECoG data (b). The structure is illustrated exemplary for mean paths of 1000 ms trials of subject MEG 2 (a) and 600 ms trials of subject ECoG 2 (cf. Figure 4.23).

single trial paths have (cf. Figure 4.22). Within this segment, approximately 1.5 to 2 states are passed through. This is about twice as much as in the LFTD datasets, even though the total amount of states is smaller (4 instead of 5). Such a rapid state change leads to higher robustness when using a threshold-based approach, as it narrows the time window within which the threshold is usually crossed. Consequently, a higher level of accuracy for duration decoding can be achieved on such data. After reaching the highest state, models dwell there for about 400 ms. Subsequently, a return to earlier states takes place. The average paths decrease approximately down to state 2, which can be interpreted as a return to the starting situation (segment I and II). This period is eventually followed by additional increases (like segment III). These correspond to further stimuli appearing within the trial segment due to the *multi stimulus issue* (Section 3.7.2: 'Multi-stimulus issue').

The duration decoding results reflect the expected outcome of these observations. Due to the robust Viterbi paths, highest duration decoding accuracy is achieved on dataset ECoG 2. Stimulus length is predicted with an RMSE of 123 ms, translating into a discrete accuracy of more than 82 %. Mean Viterbi paths of dataset ECoG 1 have the same qualitative behavior; individual structures however, are less marked. This is caused by less consistent single trial paths (cf. Figure 4.22). Differences become particularly prominent for segment III, which is less steep and covers a smaller state span (approx. 1 to 1.5 states). As a consequence, some paths reach the last state somewhere within segment IV. This decreases precision and is reflected by the significantly higher RMSE of 279 ms for that dataset. In comparison, results for the MEG datasets lay in between ECoG 1 and 2. However, since the MEG setting contains only three different durations—and these are also more widely spaced (500 ms instead of 300 ms)—the decoding task becomes simpler. In addition, the calibration step to determine the offset n uses all trials of one specific duration; this accounts for one third of the data in the MEG setting and only for one fifth in ECoG. Consequently, a more robust approximation of n can be achieved for MEG datasets, which likely also contributes to the increased accuracy of the estimation of duration in comparison to ECoG.

More detailed information on the decoding results can be taken from the error histograms (Figure 4.24) and confusion matrices (Figure 4.25). The histograms clearly show that the majority of classifications lead to a duration error of 0 ms for all four datasets. With the exception of ECoG 1, errors become less frequent with increasing error value. For dataset ECoG 1, several small peaks can be found in the histogram. These appear at positions with constant distance of 300 ms to each other, which indicates that they originate from predictions with a fixed erroneous duration. The fact that these errors range up to high values of about 1400 ms leads to the conclusion that this can be attributed to trials for which the Viterbi path never reaches the threshold. In those cases, duration prediction yields approximately[11] 1700 ms; the resulting errors are 1400 ms, 1100 ms, ..., 200 ms, which match the peak positions.

The confusion matrices (Figure 4.25) provide information on the distribution of decoding errors for discrete duration classes. Especially for the ECoG results, it becomes apparent that decoding accuracy decreases with increasing presentation duration. This is expected considering that the applied decoding strategy is based on the first time point at which the threshold state is reached. Hence, later picture offsets leave more room for premature detections than shorter ones.[12] For the two datasets with the lowest duration decoding accuracies (i. e. ECoG 1 and MEG 1), frequent mix-ups between the shortest duration classes can be found. Approximately 35-40 % of the corresponding trials are misclassified. This is most likely due to the use of a constant threshold for all time points and the issue might be addressed with more sophisticated routines for the analysis of the Viterbi paths.

[11]This results from Eq. (4.2) with $s = 100$ (last sample of Viterbi path) and an average offset of $\bar{n} = -157.4$ (cf. Table 4.7).

[12]One could expect that a similar effect appears in case of short presentation durations and delayed detections. However, this does not appear since only the first threshold crossing is considered. If the initial (early) picture offset is decoded correctly, later threshold crossings are ignored.

Such routines might contain linearly increasing thresholds for increasing duration.[13]

In case of ECoG data, Viterbi paths return to their initial state after picture offset, enabling the decoder to detect further events. This can be observed in Figure 4.22 for trials with short stimulus duration. It is expectable that the same structure would be present when using longer data segments or continuous feature streaming. Therefore, the approach is promising for a potential continuous decoding scenario.

With the presented method, calibration of the offset parameter is required. This has been done based on a single calibration class using each of the classes once for calibration in a CV-like manner. For the same feature type, calibration results show similar results across all calibration classes (cf. Table 4.7). Hence, it might be feasible to define a global offset value that can be used across different subjects without severe impact on decoding accuracy. This would provide a fully unsupervised method. However, conclusive proof of the feasibility would require evaluation on a significantly larger database, which is unavailable in this study.

Conclusion The results of this section demonstrate the use of the beneficial properties of HMM decoders to extract additional (temporal) information about a stimulus in the context of a BCI decoding problem. This is done by utilizing the features of dynamic classifiers, emphasizing their advantage over conventional static classifiers in more complex settings. The approach allows for simultaneous decoding of both a *quality* (category) and a *quantity* (duration) of an event without an additional training step for the latter. With respect to potential application, this approach might be transferred to typical BCI tasks, such as movements, e. g. by detection of desired movement direction (quality) and distance (quantity).

The approach has been demonstrated for single-trial decoding. However, for practical use of a BCI, continuous decoding output is essential. This demand can be addressed conveniently with HMM-based routines, which will be investigated more closely in the remaining result sections.

[13]Note that a rough linear trend of the position of kinks in the paths can be observed in Figure 4.23.

4.4. Semi-continuous decoding

In Section 4.3, the properties of HMMs as a dynamic classifier have been used successfully to extract additional information without a dedicated training step. There is no direct analogue to this approach using static classifiers. However, classification accuracies for the primary decoding task (i. e. picture category) turn out to be slightly lower than those of a gold-standard SVM decoder. In Section 4.3, HMMs have been applied in a straightforward single-trial decoding setting; hence, full potential of dynamic classification might not be used. This chapter focuses on the extension of the routine to a semi-continuous approach to further exploit the possibilities of HMM decoding.

Data All investigations are performed on the ECoG 1 and ECoG 2 datasets[14] from the picture category task (Sections 3.7.2 and 4.3). Trial structure, feature extraction and channel selection remain unchanged compared to Section 4.3.

Approach While the investigations is Sections 4.1–4.3 are based on the HMM toolkit for MATLAB by Kevin Murphy [115], starting from this section, the gold-standard speech recognition framework HTK (Hidden Markov Model Toolkit, [111]) is used to provide decoding functionality (cf. Section 3.5).

Instead of restricting the decoder to use only a single model to describe an entire trial, compositions of multiple models are allowed in the *semi-continuous* (SC) setting. Technically, SC decoding is realized with the word network functionality of HTK using a fully-connected word network with equal transition probabilities between all models (Figure 4.27). Different from isolated unit decoding, predictions from SC decoding can be composed of multiple different classes within an individual trial, which is illustrated schematically in Figure 4.28. Consequently, a decision needs to be made which of the predictions is considered for evaluation in order to assess decoding accuracy.

The naïve approach is to chose the model that spans the longest consecutive interval, as it best explains the most data samples. This approach will be considered as default setting for the SC scenarios (*SC default*). Using more problem-adapted strategies instead offers the option to include prior knowledge. This will be explained in the following. As described in Section 3.7.2, multiple stimuli could occur within the data segment of a single trial due to the design of the experimental paradigm (cf. top part of Figure 4.28). If such an additional stimulus is contained in a trial, it is assured that this happens towards the end of the data segment. The original stimulus is guaranteed to occur first.[15] In the *SC first word* setting, this information is incorporated by selecting the first model appearing in the prediction as the decoded label for the trial. In addition to that, it is also known that the shortest stimuli in the paradigm have a duration of 300 ms. Hence, another setting (*SC minimum length*) is investigated to analyze the potential of incorporating this

[14]The MEG datasets have not been considered here. The appearance of the multi-stimulus-issue (MSI) makes the ECoG datasets a highly interesting basis to study the possibilities of dynamic decoding, while the MEG paradigm has been designed to not contain the MSI by concept.

[15]This is more or less true by definition, since the label that is assigned to the trial is based on the category of the picture that is presented at the beginning of the data epoch (i. e. at time point 0 ms).

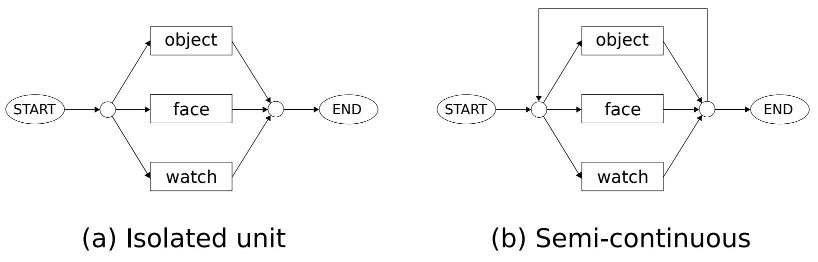


Figure 4.27: Scheme of isolated unit decoding (a) and a semi-continuous setting using a simple word network (b).

Table 4.9: Overview of the different decoding settings.

Setting	Type	Strategy
Vanilla HMM	isolated-unit	select HMM with maximum likelihood
SC default	semi-continuous	select model of longest interval
SC first word		select model of first interval
SC minimum length		first interval with more than 20 samples

information. To do so, intervals that span less than 20 samples (corresponding to 375 ms) are not considered for selection of the first interval.[16] For comparison, results are also assessed with an isolated unit decoder (*Vanilla HMM*). All decoding settings are summarized in Table 4.9 and illustrated schematically in Figure 4.28.

Results Figure 4.29 shows the resulting decoding accuracies in all investigated settings (10-times-5-fold CVs). SVM performances are taken from Section 4.3 (cf. Figure 4.20). The use of a semi-continuous approach without incorporation of any PK ('SC default' case) leads to small decreases in decoding accuracy compared to the isolated unit decoder ('Vanilla HMM'). This observation will be discussed later. All settings with PK incorporation provide substantial increases in accuracy. On average, 7.7 % more decoding accuracy (absolute increase) is achieved with the 'SC minimum length' approach with respect to 'Vanilla HMM' decoding. In both PK settings, accuracies get close to or even exceed SVM results.

To provide more insight into the results, decoding errors are studied separately for the different stimulus durations. Figure 4.30 provides a detailed overview of the decoding errors for both investigated datasets and all duration variants. It becomes apparent that errors for longer duration stimuli (i. e. 900 ms and above) increase when using the 'SC default' setting (41.9 % more errors). These errors are eliminated with the 'SC first word' setting, bringing back the error count for longer duration stimuli approximately to the level of the 'Vanilla HMM' decoder (3.7 % less errors). Shorter stimuli benefit heavily from the incorporation

[16]Note that in the unlikely event of a prediction that is composed entirely of intervals shorter than 20 samples, this routine would not yield any meaningful result.

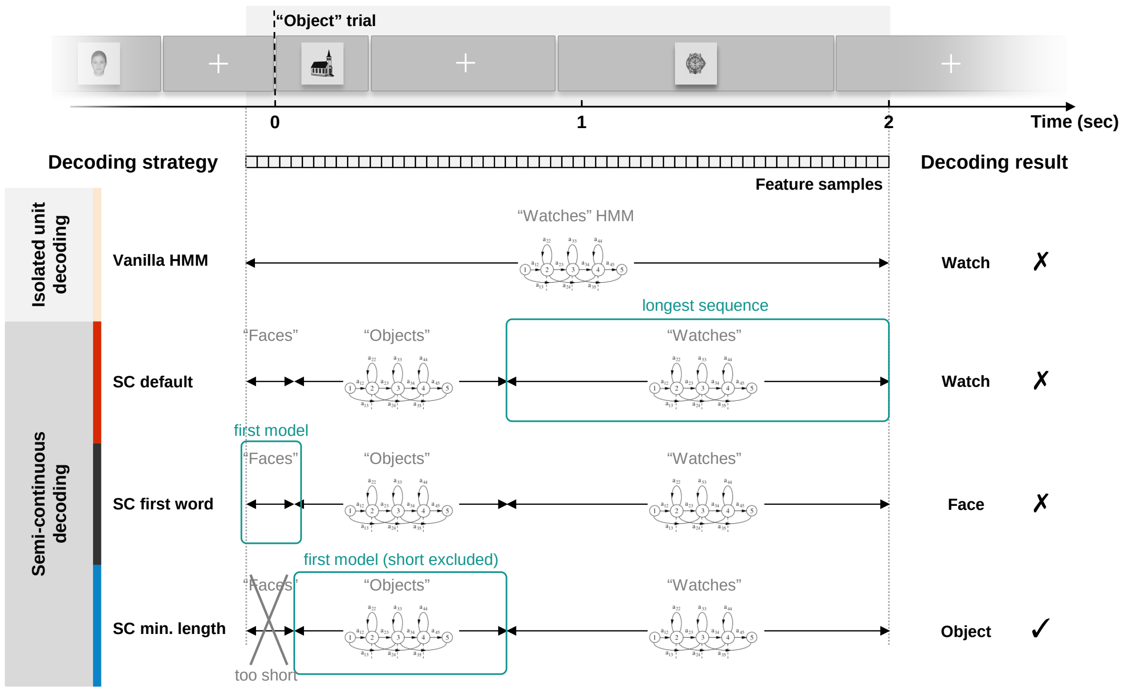


Figure 4.28: Schematic overview of the different investigated decoding settings. The *Vanilla HMM* represents an isolated unit decoder as it assigns a single label to the entire trial. In all semi-continuous (SC) settings, multiple models can occur in the most likely sequence. The presented approaches differ in the strategy on how to chose the final label for the trial when multiple models occur. In the *SC default* setting, the longest sequence determines the label, whereas the first model is chosen in the *SC first word* case. The *SC min. length* approach is based on 'SC first word' but rejects model segments that span less than 20 samples. In the given example (i. e. an object trial), only the 'SC min. length' strategy provides the correct result.

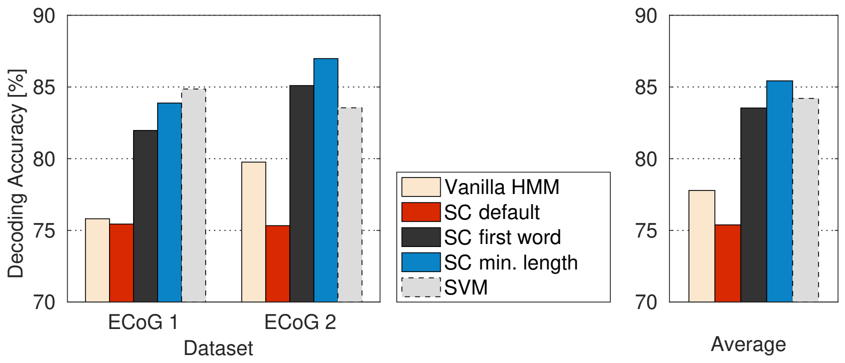
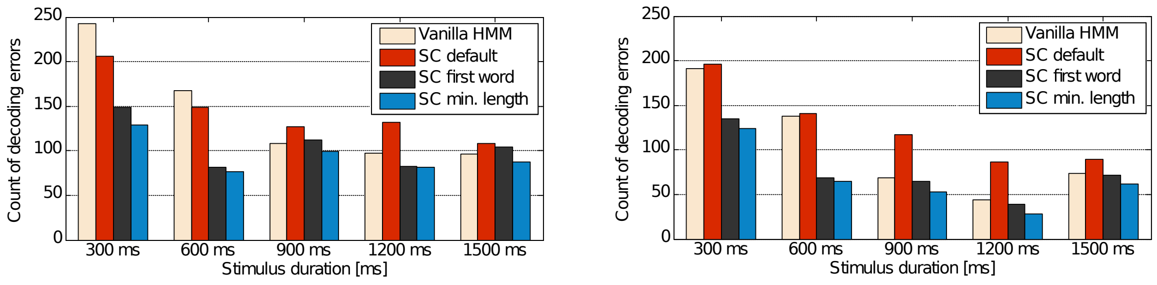


Figure 4.29: Decoding accuracies for both datasets and all decoding strategies (chance level: 33.3%; dashed gray bar: SVM results). (Figure adapted from [45], extended & recolored.)



(a) ECoG 1

(b) ECoG 2

Figure 4.30: Count of category decoding errors for both datasets and all decoding strategies. Error counts are summed up over all ten repetitions of the 10-times-5-fold CV routine and have been grouped by stimulus duration. Total count of evaluated trials in the 10-times-5-fold CV routine is 2940 (ECoG 1) and 2550 (ECoG 2). (Figure adapted from [45].)

Table 4.10: Relative change in count of decoding errors with the semi-continuous approaches compared to 'Vanilla HMM'. Results have been averaged across both datasets and across the following durations: 'Short stimuli' := $\{300\,\text{ms}, 600\,\text{ms}\}$; 'Long stimuli' := $\{900\,\text{ms}, 1200\,\text{ms}, 1500\,\text{ms}\}$.

	No PK	*With PK*	
	SC default	SC first word	SC min. length
Short stimuli	-5.3 %	-42.2 %	-47.2 %
Long stimuli	+41.9 %	-3.7 %	-18.2 %

(Positive values indicate increased error rate)

of PK. On average across both datasets, decoding errors of 300 ms and 600 ms trials are reduced by more than 42 % and 47 % with the 'SC first word' and 'SC min. length' approach, respectively. Table 4.10 summarizes the performance changes in the various settings.

Discussion The error distribution (Figure 4.30) shows that most of the decoding errors appear for short stimuli when using the isolated unit decoder ('Vanilla HMM', routine analogue to Section 4.3). This is likely due to the multi-stimulus-issue (MSI, cf. Section 3.7.2). In case of a short original stimuli that is followed by a longer additional stimulus, the HMM corresponding to the additional stimulus provides the higher likelihood for the entire trial, simply as it explains a higher number of samples more accurately. This is not addressed with semi-continuous (SC) decoding without PK incorporation ('SC default'). On the contrary, the influence of this effect on the decoding accuracy can even be intensified, as decision is always based on the longest sequence in 'SC default'.[17] Introduction of PK can compensate this and leads to substantial rises in decoding accuracies. The 'SC first word' strategy solves the problem with decoding errors that originate from the MSI (as described above) by choosing the first model in the prediction. In doing so, other models that might

[17]An example is provided in the Appendix (Section A.1) to illustrate this issue.

describe additional stimuli in later segments of the trial are ignored, no matter how high their contribution to the total probability might be. Since MSI mainly affects trials with short stimulus durations, the highest gain in decoding accuracy is expected for 300 ms and 600 ms trials. This is clearly reflected by the results.

Interestingly, use of a PK-free SC strategy ('SC default') introduces a relatively large amount of errors for longer duration stimuli. This might be attributed to the increased complexity of the decoding routine and could potentially be mitigated by introducing a model transition penalty (not investigated here). PK incorporation helps reduce these errors markedly. For the affected longer duration trials, 'SC first word' provides similar accuracy as a 'Vanilla HMM' decoder. In total, the 'SC first word' setting nearly achieves the same decoding accuracy as the SVM decoder. Performance is further improved by the introduction of the minimum length restriction. This constraint compensates for decoding errors that appear immediately in the beginning of a trial segment. Since the trials used for the study cover a time interval that already begins 100 ms before the stimulus appears, there is a possibility that the very first samples in a trial are better described by the model representing the previous stimulus. This might be interpreted as some sort of "mental afterglow" of the previous stimulus and affects trials of all durations almost equally, which can also be seen in the results. The 'SC min. length' applies the length restriction, and by that achieves an additional performance gain of 1.9 % (absolute increase, avg. across both subjects) compared to 'SC first word'.

Conclusion By combining the benefits of two components of PK incorporation, the 'SC min. length' approach provides decoding accuracies at the same level of an SVM-based decoder while still preserving the benefits of dynamic decoding. The use of semi-continuous approaches facilitates convenient incorporation of prior knowledge, providing significant increases in decoding accuracy. While the investigated setting might not represent a typical application scenario, it still provides essential insight into the possibilities of dynamic decoding of brain signals. The results strongly indicate that the approach has high potential, especially with respect to continuous decoding and more specific PK incorporation. These two aspects will be investigated in the following sections.

4.5. Continuous decoding of finger movements

The previous sections demonstrated the application of HMM-based decoding approaches for single trial classification tasks. In real-life BCI applications, however, continuous decoding output is desirable.[18] As discussed in Section 2.3.1, dynamic approaches are well-suited for this purpose. This section covers the application of an HMM-based routine for continuous decoding of finger movements.

The central findings of this section have been published in [46].

Data Data from the finger tapping experiment (Section 3.7.1)—these have been investigated on a single trial basis in Section 4.1—are reused here. This allows to build upon the findings from the previous investigations and provides a basis for comparisons. The results from Section 4.1 showed that the data from one subject (S2) is unsuited for adequate decoding of individual finger movements. As discussed, this is most likely due to insufficient coverage of corresponding motor cortex areas by the electrode grid. As a consequence, the affected datasets (i. e. S2-B1 to S2-B4) are not considered here.

Contrary to single trial decoding in Section 4.1 that was focused on separate cross-validation evaluations for each recording session, *cross-session analysis* shall be carried out here. In this setting, training and testing are performed on data from different sessions. Data were pre-examined to prove whether they are consistent across sessions, which is a requirement to meaningfully perform decoding across different sessions. To analyze this aspect, high gamma (HG, see Section 3.1.2 and 4.1) feature values $f_{c,s}$ have been averaged across all samples s of each recording session, individually for all channels c:

$$\mathrm{HG}_c^{\text{Session 1/2}} = \frac{1}{S}\sum_{s=1}^{S} f_{c,s}^{\text{Session 1/2}},\ c = 1, ..., C.$$

Values of both sessions are compared by computing their ratio

$$r_c = \frac{\mathrm{HG}_c^{\text{Session 1}}}{\mathrm{HG}_c^{\text{Session 2}}},\ c = 1, ..., C.$$

This ratio is expected to have a value around 1.0 if data are consistent across sessions. Figure 4.31 shows r_c for all C channels in the datasets. Apparently, data from subject S3 are inconsistent as they show highly varying HG ratios of $1 \lesssim r_c < 2.5$. To compensate for this—and thereby, condition the data for use in cross-session analysis—channels of session 1 have been rescaled individually to match the average value of session 2:

$$\tilde{f}_{c,s}^{\text{Session 1}} = \frac{1}{r_c} \cdot f_{c,s}^{\text{Session 1}}, \forall c, s.$$

[18]This is due to the fact that single trial settings are artificial. User intentions typically do not occur at well-defined (or even equidistant) time points. Instead, it is likely that individual trials will contain multiple events (or overlaps) in practice due to varying speed of execution. Consequently, single trial approaches limit the possible dynamics of such systems, and can pose severe problems for decoding.

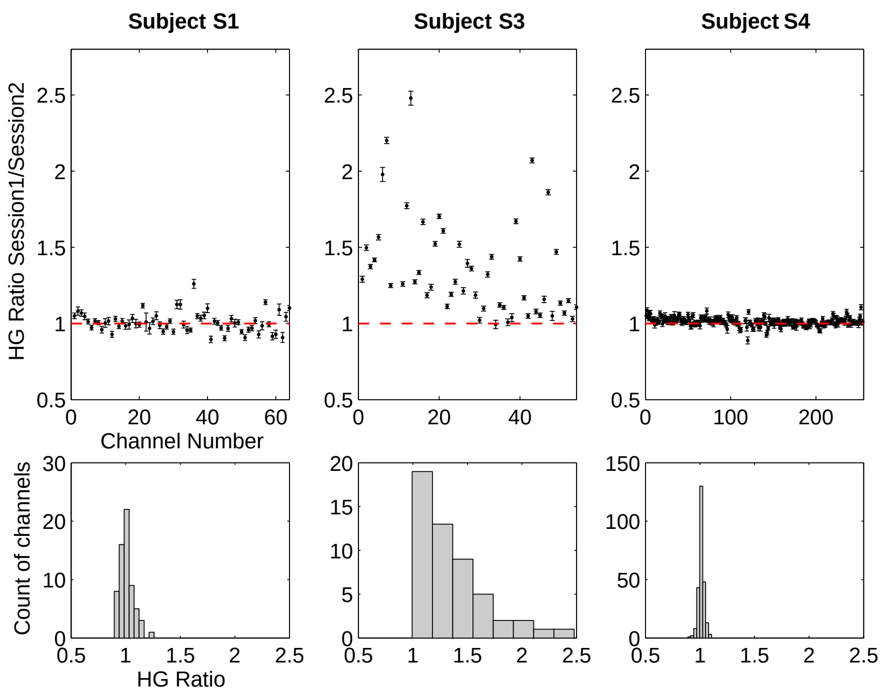


Figure 4.31: HG feature ratio between session 1 and session 2 for all three subjects (upper row). Markers show the ratio for all individual channels (error bars indicate standard error). In case of consistency across sessions, data are expected to have a ratio around 1.0 (indicated by dashed red line). The count of channels with particular HG ratios are illustrated with histograms (bottom row). A Gaussian distribution (typical noise characteristics) in the histogram would be expected for consistent data. Apparently, data from subject S3 are inconsistent across the two recorded sessions. Since HG ratio is different for all channels (top row, middle chart), compensation of the inconsistencies by multiplication with a common factor is not possible. Hence, channels need to be rescaled individually. (Figure reproduced from [46].)

Approach The HMM decoder is realized with the gold-standard speech decoding framework *Hidden Markov Model Toolkit* (HTK). A C++ wrapper has been implemented (Section 3.5) to manage the data and convert it into HTK compatible format. Furthermore, model initialization routines and wrapper methods for all required HTK sub-routines have been implemented. The following HTK routines are used:

HRest	basic Baum-Welch re-estimation of the parameters of a single HMM using a set of observation sequences (model training) [126].
HVite	general-purpose Viterbi word recognizer, which matches a given feature sequence against a set of HMMs and outputs a prediction [126].

In addition to the single trial case, resting state is also decoded in the continuous setting. Hence, a total of five HMMs are trained: four for the finger movements (thumb, index finger, middle finger, and little finger) and one HMM for the resting condition (Figure 4.32). The finger movement models are defined to have three states in a left-to-right topology. This means that model transitions are restricted to loops and single-stepped forward

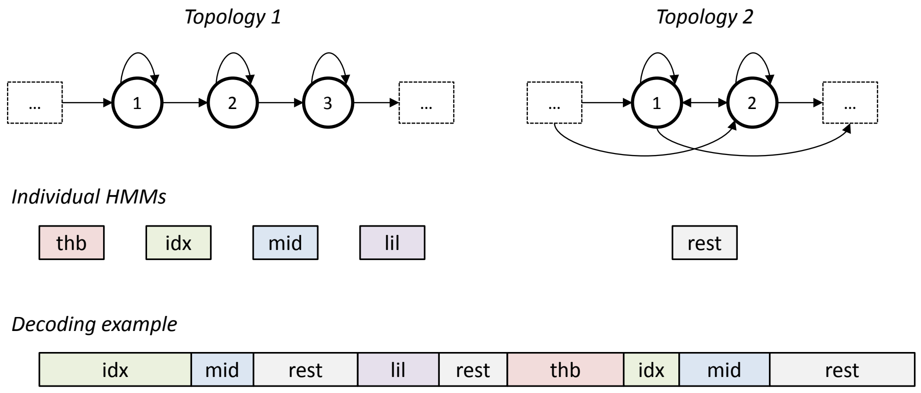


Figure 4.32: Topology of HMMs associated with finger movements ('Topology 1', left) and resting condition ('Topology 2', right). The dashed rectangles represent the interconnection from/to the previous/following HMM, respectively. (Figure adapted from [46].)

jumps. Neither backward jumps nor state skips are permitted. All states have emission probability distributions modeled with a single Gaussian mixture component and diagonal covariance matrices. For the resting condition, a two-state fully-connected topology with full covariance matrices is used. Note that this topology is equivalent to an HMM with a single state using two Gaussian mixture components. However, the two-state variant is chosen instead for technical reasons, as it allows to process all HMMs in a structurally equal manner within the framework.

The use of an alternative model structure for the resting condition is motivated by the specific properties that are assumed for this data. It is to be expected that rest episodes do not contain any reproducible dynamics, contrary to actual movements that may potentially follow a certain temporal pattern (e. g. planning, execution, and "cool-down" phase). Hence, modeling the temporal structure of resting periods is expected to be less meaningful. This is reflected by the reduced state count and a more fuzzy connectivity setting (fully-connected states). In the absence of movements, no profound assumptions on the correlations between neural activity from different spatial locations can be made. Thus, full covariance matrices are used to model emission probabilities for the rest HMM. It should be noted that in ASR, it is usually sufficient to use a single state HMM for silence modeling. However, empirical findings reveal that the two-state model generalizes better across sessions and subjects (not shown here).

As indicated above, evaluation is done in cross-session structure (Figure 4.33). This means that one session is used to train the decoder and determine all required parameters; the trained model is then used to decode the remaining session. This is done in both directions, i. e. using session 1 for training and session 2 for testing (referred to as 'B1B2') and the other way around ('B2B1').

Using the training set, individual training segments are prepared following the same principle as in the single trial analysis. Data are epoched into time segments that cover the interval of [-100, 400] ms around a button press. Pause episodes in the data are used to construct training segments for the rest model. All consecutive intervals of more than

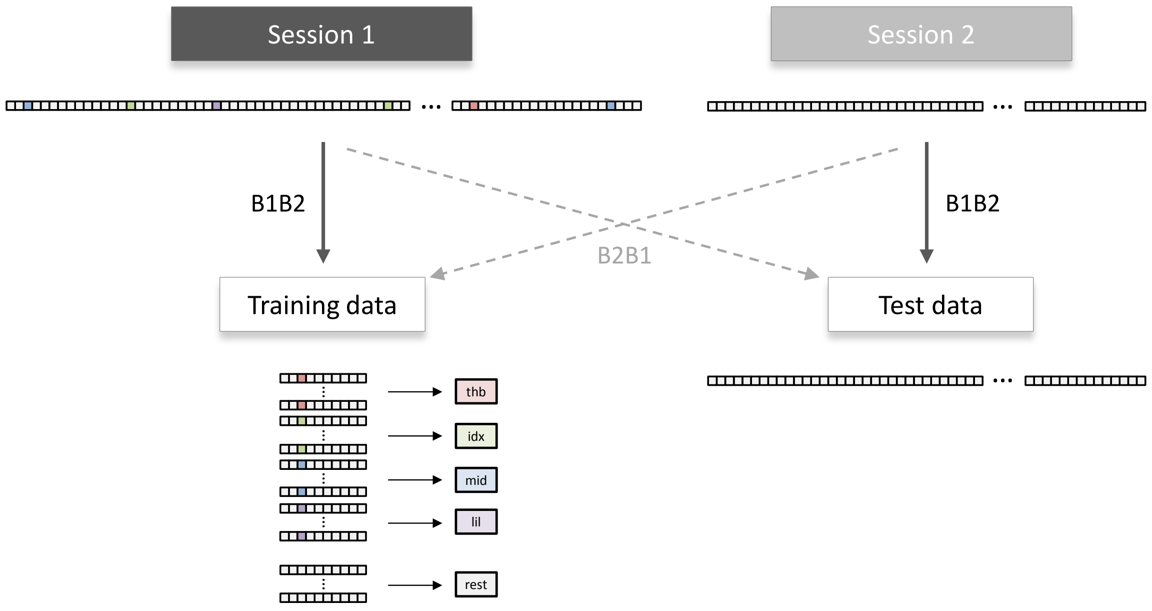


Figure 4.33: Scheme of the cross-session evaluation structure. Data from one session are used to train the decoder which is then evaluated on data from the other session. Settings in which session 1 is used for training and session 2 for testing are denoted with the suffix '-B1B2' (other direction analogously '-B2B1'). The small rectangles symbolize individual data samples. (Figure reproduced from [46].)

ten seconds without any button press are considered as pauses. These data are epoched into non-overlapping segments with a length of 500 ms (identical to movement segments). In advance, all raw data have been reviewed manually for artifacts (mainly due to epileptic activity). The affected time intervals are not considered for training. Table 4.11 shows a breakdown of the total duration and the accumulated length of the contained pause and artifact intervals for all datasets as well as the total count of events (per movement type) that occur in the dataset.

The experiment was structured in three *sub-blocks* (SB) with varying task for the subject (cf. Section 3.7.1). Table 4.12 provides an overview over the number of button presses in each of the sub-blocks. As mentioned in Section 3.7.1, the average frequency of button presses differs across the sub-blocks. In particular for SB3, a significant increase in tapping speed is expected. To provide evidence for this assumption, the average[19] time intervals between consecutive button presses have been assessed (Table 4.12). On average, tapping speed in SB3 is approximately 25 % higher than in the other two sub-blocks. For individual datasets (esp. S1-B1, S4-B2), this increase is even more dramatic. To ensure equal conditions for all evaluations, data from SB3 are not used for decoder training. Table 4.13 summarizes the resulting count of training segments for movements and rest for all datasets.

Models are initialized with the time-discriminative k-means clustering routine that has also been used in single trial decoding of this data (see Section 4.1). The time coupling parameter has been set to $\tau = 5$. As described in Section 3.3.2, the clustering step requires

[19]The median has been computed instead of simple arithmetic average to compensate for outliers.

Table 4.11: Duration of measured ECoG data, pauses and artifact episodes as well as a breakdown on all button press events in the recordings.

		Duration [s]			Count of events				
Subject	Session	Total	Pauses	Artifacts	THB	IDX	MID	LIL	Sum
S1	B1	490	94	0	61	116	107	75	359
	B2	299	34.75	0	35	92	71	45	243
S3	B1	703	151	173	51	77	72	49	249
	B2	591	107.75	65	51	80	96	59	286
S4	B1	442	86	0	56	112	114	74	356
	B2	421	85	49.5	61	102	105	51	319

Table 4.12: Number of movement segments (trials) after artifact rejection and median of the time interval between consecutive button presses (pause episodes left out) for all three subjects and sub-blocks. Dataset notation S1-B2 means session 2 of subject 1.

	Number of segments Sub-block *Sequence type*			Time between events [ms] Sub-block		
Dataset	SB1 *fixed*	SB2 *random*	SB3 *free choice*	SB1	SB2	SB3
S1-B1	179	60	120	1182	1100	613
S1-B2	120	123	-	941	1071	-
S3-B1	88	124	37	1468	1323	1418
S3-B2	87	149	50	1088	1264	994
S4-B1	117	180	59	863	1026	668
S4-B2	107	154	58	919	984	406
			Mean	**1076.8**	**1128.0**	**820.0**

Table 4.13: Count of training segments for all datasets. The count of movement segments is the sum of all four types of finger movements (i. e. thumb, index, middle and little finger).

Subject	Session	Movement	Rest	Ratio Rest/Mov. [%]
S1	B1	239	183	76.6
	B2	243	59	24.3
S3	B1	212	108	50.9
	B2	236	136	57.6
S4	B1	297	144	48.5
	B2	261	98	37.5

an initialization that is based on a routine involving (pseudo-)random values. In order to reduce the influence of the initialization on the final results, all evaluations are repeated ten times with different seeds for the random number generator and results are averaged across all ten runs. Channel selection has also been performed using the same routines as in the single trial analysis. For all datasets, the ten most informative channels have been selected based on the training set. After initialization, HMM parameters have been re-estimated using ten iterations of a Baum–Welch routine, as provided by the HTK sub-routine 'HRest'.

Continuous recognition of finger movements is performed with a Viterbi-based algorithm using the HTK tool 'HVite'. The sensitivity of the decoder can be controlled with a penalty term that diminishes the likelihood for a sequence each time a new detection starts (i. e. with every transition between models). Due to its origin in ASR, this term will be referred to as *word insertion penalty* p. Its effect on the decoding output is that the word insertion penalty controls the balance between false positives and false negatives. Low values of p typically result in high false positive rates, while too high penalties lead to extensive false negative rates. The optimal value for p is determined by exhaustive search on the training set.

In order to evaluate the accuracy of the decoding output, a comparison routine to match the predictions against the ground-truth as well as a performance measure are required. These components are explained in detail in Section 3.6.2.

Results By definition, the applied evaluation routine has two parameters: *offset* and *tolerance* (cf. Section 3.6.2). Both parameters influence the performance score of a given prediction. Since this is crucial for all subsequent analysis, their influence is investigated first. The offset is introduced to compensate for shifts between the prediction and the actual events. Generally, the training set could be used to determine the offset value that maximizes performance. This value is likely to constitute an estimate that is suitable for evaluation of the test set. Figure 4.34 (a) shows the performance for all dataset with varying offset parameter. It becomes apparent that decoding performances reach a distinct peak at a certain offset value. The peak position is similar for all datasets[20], indicating that a common value could be used instead. The average across all datasets has its peak at 40 samples. Hence, this value is used across all datasets.

Different from the offset parameter that explicitly describes an underlying effect of the data (i. e. temporal shift), the tolerance parameter is entirely a technical instrument used to assess meaningful results. As described in Section 3.6.2, an optimal value for the tolerance parameter should provide a compromise between strictness of evaluation (i. e. tolerance as low as possible) and coverage of fluctuations in the results. This is important to allow fair comparisons between different datasets. Figure 4.34 (b) shows the relationship between tolerance and resulting decoding performance. It can be clearly seen that all curves have identical qualitative trend. For all datasets, performance reaches an approximately constant

[20]The only outlier is S4-B2B2 (light blue curve in Figure 4.34 (a)). However, a closer look reveals that the peak performance for this dataset forms an extended plateau. Therefore, the performance difference between the peak point at offset 32 samples and, for example, an offset of 39 samples is less than 0.6 %. This brings that dataset in line with all other datasets.

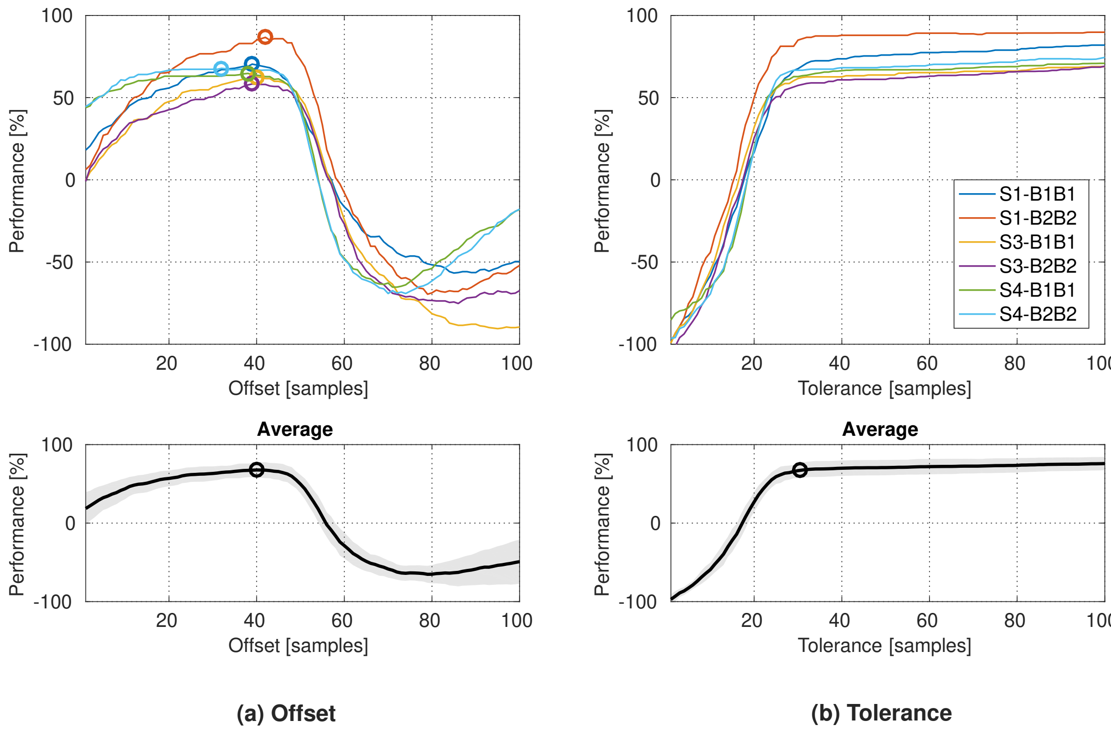
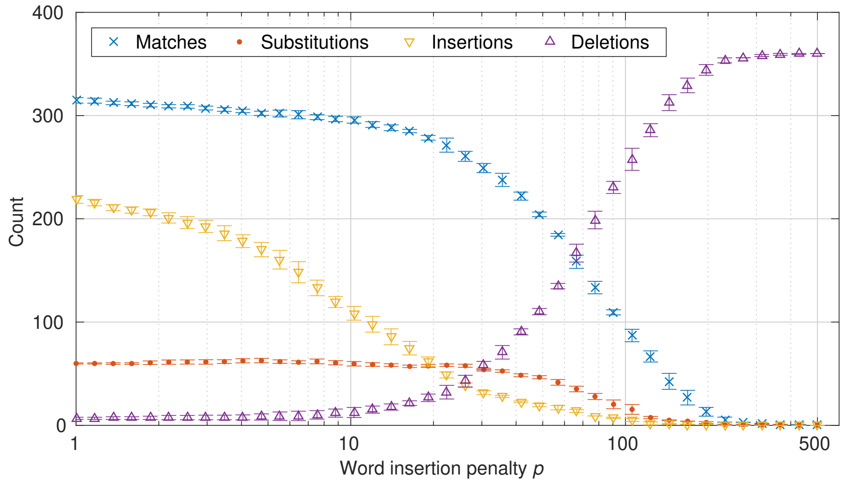


Figure 4.34: Influence of offset (a) and tolerance (b) parameter on the decoding accuracy. To compute results for the offset dependency (a), a fixed value of 30 samples tolerance has been used (tolerance results (b): 40 samples offset). Graphs are shown for all individual datasets (top row). Since this procedure is used to determine the optimal settings for decoding, training and testing are performed on same dataset. The optimal offset value (i. e. performance peak) is highlighted with a circle marker. In addition to individual results, mean across all datasets is shown in the bottom row (shaded area indicates standard deviation). Here, the circle marker indicates the final choice of both parameters that is used throughout all further analysis. (Sub-figures in the bottom row are adapted from [46].)

level for tolerance values of 30 samples and above. This can also be taken from the averaged curve in the bottom of Figure 4.34 (b). Consequently, a tolerance of 30 samples is used for evaluation.

As mentioned before, decoder sensitivity can be controlled with a penalty term. Exemplary, this will be discussed based on decoding scenario S4-B2B1. Figure 4.35 shows the dependency of matching predictions and all types of decoding errors on the word insertion penalty p. All results have been averaged across ten repetitions with varying seed for model initialization. As expected, increasing penalties lead to a reduction in the total amount of detections, which is reflected by a reduced count of matches, substitutions, and insertions along with a corresponding increase in the count of deletions. Using Eq. (3.3), *pseudo-performance* values (simply referred to as *performance* in the following) have been computed from each of the results. The evaluation shows that—in this example—the performance P is highest for a penalty value of $p = 22.4$. In all real cross-session decoding settings, the penalty parameter is chosen based on training data only. For the presented



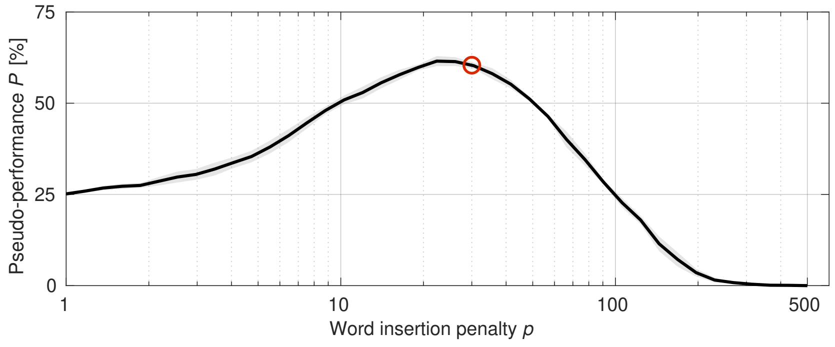
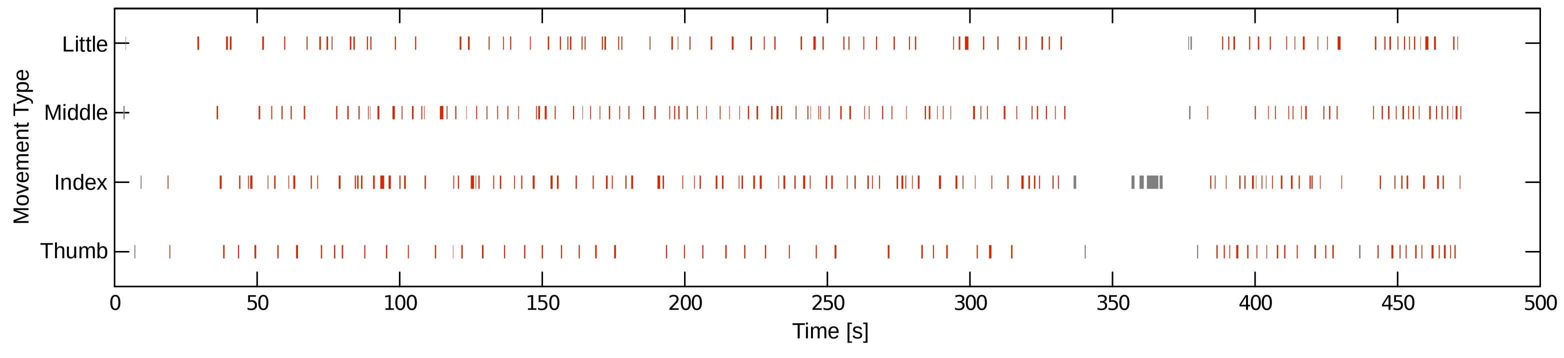


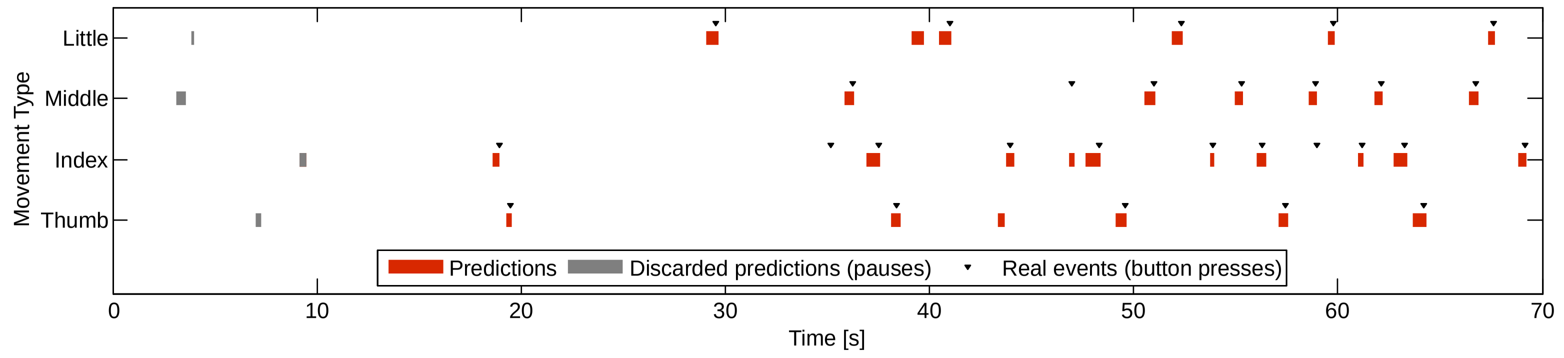
Figure 4.35: Influence of word insertion penalty on decoder sensitivity: (top) Count of matches and decoding errors (substitutions, insertions, and deletions) with varying word insertion penalty for decoding of S4-B2B1. Results have been averaged across ten runs with varying seed for the random number generation used in model initialization (error bars indicate standard deviation). (bottom) Resulting pseudo-performance values. The shaded area indicates standard deviation. The red circle shows the penalty value that is chosen based on optimization on training data.

case of S4-B2B1, exhaustive search on the training data (i. e. S4-B1B1) yields an optimal value of $p = 30$ (indicated as red circle in Figure 4.35, bottom chart). Note that the resulting performance with $p = 30$ is only about 1.3 % lower than with $p = 22.4$, which would be the result of optimization on test data.

Optimal penalty values are assessed individually for all datasets using the corresponding training set. Penalty values and the resulting decoding performances for all datasets are documented in Table 4.14. The average performance across all datasets is $\bar{P} = 53.5\,\%$. Figure 4.36 (a) shows an example of a full prediction output of the HMM decoder from S1-B2B1 decoding (i. e. training on session B2 and evaluation on B1). To facilitate visibility of details, Figure 4.36 (b) additionally provides a zoomed view of the first 70 seconds of these results. For computation of decoding accuracies, all predictions that fall within pauses during the experiment or episodes of epileptic activity have been discarded (indicated with gray color in Figure 4.36).



(a)



(b)

Figure 4.36: (a) Continuous prediction output for S1-B2B1 decoding (full detail available at higher zoom levels). Predictions that fall within pause episodes are shown in gray color. These are discarded for evaluation. Prediction runs until second 490. (Figure reproduced from [46].)
(b) Zoomed view of the first 70 seconds of the prediction shown in (a). Like in the full results, pause episodes are shown in gray color. For comparison, the time points at which the subject actually pressed buttons are indicated with black triangles. (Figure reproduced from [46].)

Table 4.14: Continuous decoding performance P and word insertion penalties p for all datasets. Performance values have been computed by means of Eq. (3.3) as the average across ten runs with differing seeds for model initialization. Corresponding standard deviations are listed in brackets.

					Errors		
Subject	Session	p	P [%] (std.)	Matches / Events	Subst.	Ins.	Del.
S1	B1B2	41	66.96 (1.48)	179.6 / 243	10.4	18.9	50.0
	B2B1	56	73.41 (1.05)	309.2 / 359	9.0	45.0	41.7
S3	B1B2	35	40.65 (1.59)	211.7 / 286	46.5	79.2	67.7
	B2B1	30	36.18 (1.39)	198.6 / 249	66.3	88.5	39.4
S4	B1B2	22	43.68 (1.37)	206.1 / 319	63.3	53.0	81.1
	B2B1	30	60.25 (0.60)	249.7 / 356	54.4	32.2	56.9

Discussion The investigation of the influence of the offset shows that almost the same value turns out to be optimal for all datasets (cf. Figure 4.34 (a)). This is an expected result, due to the fact that a significant contribution to the shift between prediction and ground truth is introduced artificially by the properties of the applied feature extraction and model training routines, which will be explained in the following: The computation of HG features uses a window length of 250 ms. In all analysis documented here, the starting point of such a window is associated with the current time point of the feature. Note that this is an arbitrary choice and it might also be reasonable to use the center or the end of the window instead. Selecting the starting point means that the resulting HG features "look into the future". Hence, this causes the decoder output to precede the actual event by approximately the length of the window used for computation of the feature value. In this particular case, this translates to an expected offset of 25 samples ($250\,\text{ms} \triangleq 25\,\text{samples} \cdot 10\,\text{ms}\,\text{shift/sample}$). It is important to recognize that this does not mean that the decoder can predict events before they actually happen, as the computation of the required feature value can only be performed as soon as the full length of 250 ms of data are acquired. Consequently, this component of the offset is entirely artificial, caused by the choice of reference point for the feature values.

Besides feature extraction, the applied model training routine also contributes to the offset. All HMMs are trained on data segments that begin 100 ms before the actual button press. Hence, the onset of a detection is expected to coincide with the time point 100 ms in advance of the button press. This results in an additional offset of approximately 10 samples ($100\,\text{ms} \triangleq 10\,\text{samples} \cdot 10\,\text{ms}\,\text{shift/sample}$).[21] In total, a shift of 35 samples between prediction and ground truth can be attributed to technical sources (feature extraction and model training). The remaining difference of five samples to the chosen offset of 40 samples might be explained by delays between brain responses and actual movements. As it is

[21]Note that this obviously cannot hold true for arbitrarily long time intervals—thus, for example, training with data that begins several seconds before the actual button press will certainly not mean that movement detections will start at that point. The observed shift effect might also be explained by some other aspects that are discussed in the following.

inevitable that movements start several samples before the corresponding button press is recorded—simply because the finger needs to travel a certain distance to trigger the button—it is highly plausible that prediction of movement onset is possible already at that stage. Additionally, movement planning might also be covered by the ECoG recordings, potentially providing information at an even earlier time point. However, due to the lack of ground-truth information on the actual movements (e. g. by means of video recordings), conclusive in-depth analysis of these aspects is not feasible.

To provide fair comparisons between individual datasets, a carefully selected tolerance value is essential. Figure 4.34 (b) reveals that the influence of the tolerance is qualitatively identical for all datasets. Increasing the tolerance parameter results in higher performance values as it widens the window that will be scanned for matching predictions. This holds true up to a certain point, starting from which the performance increases only marginally with higher tolerances. At this point, the tolerance window is big enough to cover the majority of fluctuations in timing of the prediction (i. e. performance saturates). All further increases in accuracy occurring at even higher tolerances are likely to not correspond to correct predictions that are just slightly out of timing, but to spurious detections that are only considered to be correct due to the extensively large tolerance. Consequently, the bend in the graphs is regarded to represent an ideal working point. From the averaged curve, a tolerance of 30 samples has been identified as an appropriate setting.

Continuous decoding accuracies vary from 36 % to 73 % across subjects. Strong variation is also observed between both sessions of subject S4. These findings are somewhat surprising, given that single trial decoding results are much closer together for those datasets (Table 4.15). This becomes particularly evident in the case of subject S3. While single trial decoding accuracies are second highest for S3, the lowest performance values among all datasets are achieved in the continuous setting. A similar drop in performance is also present for S4, albeit to a slightly lesser extent. There are several aspects that contribute to this outcome. First of all, it must be noted that, in general, continuous prediction poses a significantly more challenging decoding task. Not only is the problem extended from four classes to five classes (now including rest), but also made more complex, as additional possibilities for decoding errors come into play (i. e. insertions, deletions, timing mismatches etc.). Hence, expected performances are lower than for the single trial equivalent. Besides the increase in difficulty for the decoding task itself, decoder training is more ambitious due to the cross-session approach. In this setting, training is performed using data from a different session, which has been recorded separately. If any data properties changed from one session to the other, this may heavily impair the quality of the trained model for prediction. Such changes can be of various nature, including hard factors like electrode impedance or altered recording settings, but also—and probably more importantly—soft factors such as the subject getting used to the experimental task, influence of epileptic episodes, or simply varying motivation of the subject. It might be legitimate to assume that those factors are constant throughout the same recording session. However, this assumption might be violated across different sessions.

Another aspect that needs to be discussed is the validity of rest episode labeling. Since no physical monitoring of actual movements of the subjects (e. g. by means of data gloves

Table 4.15: Comparison of decoding accuracies from single trial (Acc.) and continuous decoding (P) for all datasets. Results of single trial decoding are taken from Section 4.1 (Table 4.4, HG features).

Subject	Session	*Single trial* Acc. [%] (std.)	*Continuous* P [%] (std.)
S1	1	97.9 (0.6)	73.4 (1.1)
	2	92.4 (2.0)	67.0 (1.5)
S2	all	left out	
S3	1	85.1 (2.2)	36.2 (1.4)
	2	90.5 (1.8)	40.7 (1.6)
S4	1	83.5 (2.7)	60.3 (0.6)
	2	83.1 (1.9)	43.7 (1.4)

or video recordings) is available, data episodes are categorized as rest solely based on the absence of button presses for a certain time period. With this strategy, it cannot be guaranteed that no other movements, like stretching or relaxing of muscles, have been performed during these segments. As a consequence, it is quite likely that some of the segments marked as rest are spoiled with movements. This impedes proper training of the rest model, and decoding performance deteriorates accordingly. To partly limit the influence of unfitting rest segments on the analysis, rest episodes are omitted in performance evaluations. Otherwise, this could strongly bias parameter selection by means of exhaustive search on training data. The example of word insertion penalty optimization illustrates the issue: If movements have been performed during segments labeled as rest, there is a high possibility that the decoder will predict movements therein. Since the segment was labeled as rest, these predictions are considered to be erroneous, which results in a significant amount of insertions. When the count of these insertions gets too high, better overall performance values are achieved by increasing the word insertion penalty. While this indeed reduces insertions, it also raises the count of deletions in the actual movement episodes at the same time. As there is no evidence that movement predictions within pause episodes are actually wrong—because it is entirely possible that there have been movements without button presses—penalizing those predictions is unjustified. To avoid this issue, word insertion penalties are selected based on optimization that excludes pause episodes from the training data. For the same reason, actual decoding performances (i. e. in the cross-session setting) have been reported for all data samples except for the (hypothetical) rest episodes. For means of completeness, results including the rest periods are documented in the Appendix (Table A.2 and A.3).

Conclusion Continuous decoding of brief finger movements can be performed with reasonable accuracy using an HMM-based approach. However, comparison to single trial results reveals that performance drops cannot be neglected. It has been discussed that a considerable amount of this reduction in accuracy can be attributed to the increased level of difficulty. The following section focuses on possibilities to compensate for the increased difficulty by incorporation of prior knowledge, and provides insight on how this can be performed in an HMM-based decoding approach.

4.6. Prior knowledge incorporation with bi-gram models

The central findings of this section have been published in [46].

Data This section is based on the data as described in Section 4.5. Full information on the datasets can be found in the corresponding tables (Table 4.11, 4.12, and 4.13). In addition, the amount of prior information (w. r. t. bi-gram frequencies) contained in the individual data sub-sessions has been evaluated using Eq. (3.4), as explained in Section 3.6.3. The resulting entropy values are shown in Table 4.16. For comparison, entropy values of typical exemplary cases are given in the following:

- Perfect SB1
$$(f_{ij})^{\mathrm{SB1}} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0.5 & 0 & 0.5 & 0 \\ 0 & 0.5 & 0 & 0.5 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$
$\Rightarrow S(\mathbf{f}^{\mathrm{SB1}}) = 2\ln(2) \approx 1.386.$

- Random used in the experimental paradigm (SB2)
$$(f_{ij})^{\mathrm{SB2}} = \begin{pmatrix} 0 & 0.\bar{3} & 0.\bar{3} & 0.\bar{3} \\ 0.\bar{3} & 0 & 0.\bar{3} & 0.\bar{3} \\ 0.\bar{3} & 0.\bar{3} & 0 & 0.\bar{3} \\ 0.\bar{3} & 0.\bar{3} & 0.\bar{3} & 0 \end{pmatrix}$$
$\Rightarrow S(\mathbf{f}^{\mathrm{SB2}}) = 4\ln(3) \approx 4.394.$

- Fully randomized sequence
$$(f_{ij})^{\mathrm{random}} = \begin{pmatrix} 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \end{pmatrix}$$
$\Rightarrow S(\mathbf{f}^{\mathrm{random}}) \overset{(3.5)}{=} S^{\max}(4) \approx 5.545.$

Approach Compared to the previous section, the decoding approach remains unaltered with respect to HMM topologies, model initialization and training as well as HTK wrapper methods. However, the framework is extended by the possibility to incorporate prior knowledge (PK) into the decoding. In particular, two different aspects of available information are considered in the approach, namely:

1. Strict alternation between movement and rest,
2. Frequencies of pairs of consecutive finger movements.

The first component—henceforth referred to as *movement–rest alternation* (MRalt)—is realized by modifying the dictionary from which the decoder constructs the prediction. In the previous section, the dictionary contained the four movement models and the rest

Table 4.16: PK information content by means of bi-gram entropy S computed using Eq. (3.4). Higher entropy values indicate less information content. Due to the fixed tapping sequence, information content is highest for SB1.

Dataset	Full dataset	Sub-block *Sequence type* SB1 *fixed*	SB2 *random*	SB3 *free choice*
S1 - B1	4.09	2.34	3.66	4.13
S1 - B2	3.94	2.68	4.39	N/A
S3 - B1	4.14	1.68	4.34	3.16
S3 - B2	3.89	1.39	4.16	3.68
S4 - B1	3.82	1.39	4.01	3.86
S4 - B2	4.08	1.39	3.89	5.00
Mean	3.99	1.81	4.08	3.97

model as individual words. Hence, two consecutive predictions could be of the same type, i. e. two movements or two rest segments. By construction of combined words that consist of the actual movement model followed by the rest model, the dictionary can be reduced to four entries of so-called "movement words". As the decoder can only predict entities from the dictionary, it is ensured that consecutive individual models are alternating between movement and rest. The construction of the combined words, along with an example of how a corresponding continuous prediction may look like, is illustrated in Figure 4.37.

To address the second component of PK—henceforth referred to as *Bi-Gram*—bi-gram models are used. These can be considered conveniently within HTK's decoding routine 'HVite'. Technically, the bi-gram model is realized by use of the word network functionality.[22] The network used to serve this purpose is shown in Figure 4.38. The network is constructed to reward transitions from one movement to another based on the relative frequency of appearance f_{ij} of the corresponding movement pair (movement of class i followed by class j). The strength of PK incorporation is controlled with a *scaling parameter* s:

$$\log P_{ij}^{\text{PK}} := f_{ij} \cdot s.$$

Scaling the PK component with $s = 0$ leads to the unbiased decoder that does not incorporate any PK on frequencies of movement sequences. Contrary, increasing values of s result in stronger influence of the bi-gram model on the decoding result while the actual observation probabilities of the underlying data become less important. In other words, the decoder more and more tends to generate the movement sequence that corresponds to the most likely transitions in the bi-gram model. Since model transitions are always rewarded in the given setting, this leads to considerable increase in false positive detections (i. e. insertions). To compensate for this, a (global) *rebalancing term* is introduced that constitutes a penalty applied for every new detection (in addition to the word insertion

[22] An example of such a word network can be found in the Appendix: Section A.3.

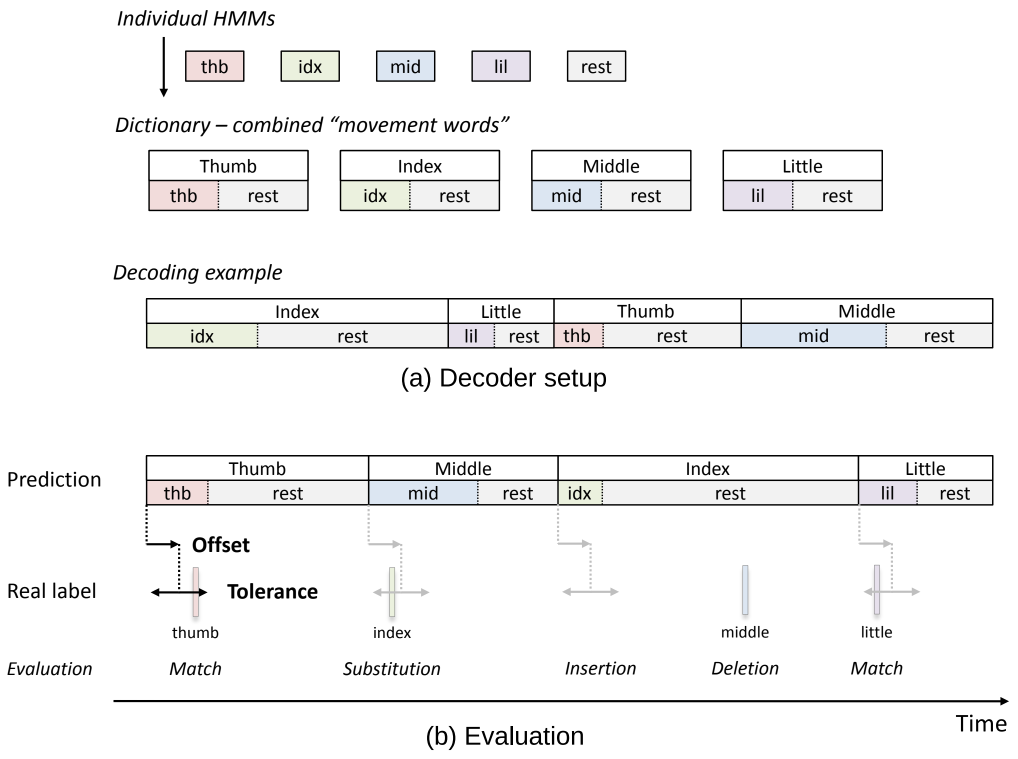


Figure 4.37: Schematic overview of decoding setup (a) and evaluation (b) for continuous decoding with PK incorporation. (a) Individual HMMs are fused into combined items consisting of a movement model and the rest HMM. The resulting four "movement words" constitute the dictionary used for decoding. The decoder can only predict elements from this dictionary. Consequently, strict alternation between movement and rest is guaranteed (cf. 'Decoding example'). (b) For evaluation, the start samples of movement HMMs are compared to the ground truth. Details on the routine can be found in Section 3.6. (Figure adapted from [46].)

penalty p). The rebalancing term is chosen to be correlated to the strength of the bi-gram model. Similar to the PK incorporation, it can also be controlled by a parameter r:

$$\log P^{\text{reweight}} := -r \cdot s.$$

Both terms have been defined to be additive corrections to the log-likelihood of the decoded sequence. The final likelihood of a decoded sequence computes as:

$$\begin{aligned}
\log P^{\text{final}} &= \log\left(P_i^{\text{HMM}} \cdot P_{ij}^{\text{PK}} \cdot P^{\text{reweight}} \cdot P_j^{\text{HMM}} \cdot P^{\text{word ins.}}\right) \\
&= \log P_i^{\text{HMM}} + \log P_{ij}^{\text{PK}} + \log P^{\text{reweight}} + \log P_j^{\text{HMM}} + \log P^{\text{word ins.}} \\
&= \underbrace{\log P_i^{\text{HMM}} + \log P_j^{\text{HMM}}}_{\text{output probabilities}} + \underbrace{s \cdot f_{ij}}_{\text{PK reward}} \underbrace{-s \cdot r - p}_{\text{penalties}}. \qquad (4.4)
\end{aligned}$$

Note that Eq. (4.4) is a significantly simplified version of the actual probability. As the full formula—which would include considering all data samples and model transitions for

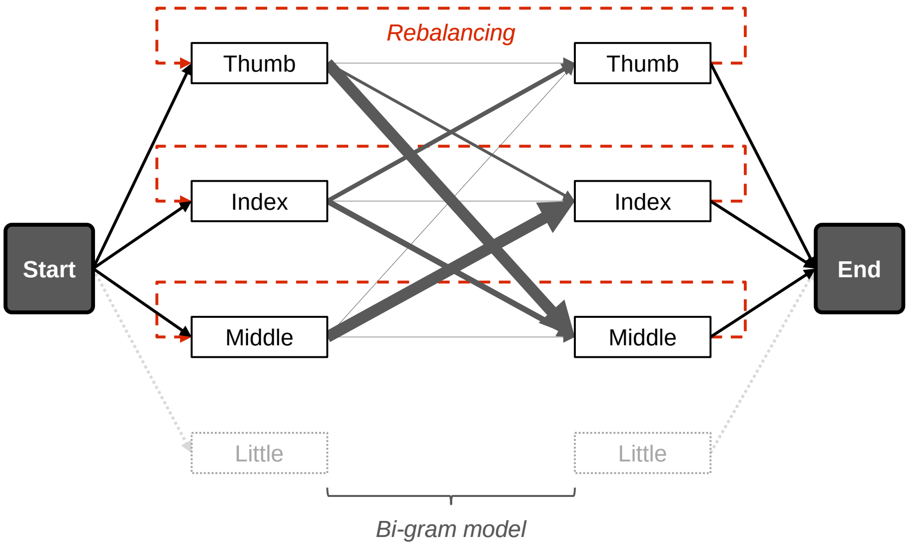


Figure 4.38: Word network used for integration of prior knowledge by means of bi-gram models into the decoding stage. The initial detection is performed without any bias from PK. Subsequent predictions are rewarded with respect to the relative frequency of appearance of the corresponding movements. To prevent extensive false-positive rates, additional detections are penalized by a rebalancing term. Note that individual elements in this scheme (e. g. 'Index') refer to "Movement words" (cf. Figure 4.37) that are a composition of the corresponding finger HMM and the rest model. For reasons of clarity, connections from/to 'Little' are omitted in the scheme. (Figure reproduced from [46].)

all possible combinations—is cumbersome, and would not provide any additional insight into the concept, the approach is presented in this way instead. Equation (4.4) describes the likelihood of a sequence that is modeled by only two HMMs, i and j. The resulting likelihood consists of

- the individual likelihood terms for both of the models (P_i^{HMM} and P_j^{HMM}),
- the PK term that rewards the transition from HMM i to j (P_{ij}^{PK}),
- the rebalancing term (P^{reweight}),
- and the word insertion penalty for insertion of the new detection 'HMM j' ($P^{\text{word ins.}}$).

In summary, the incorporation of PK with the presented approach depends on three factors:

1. Frequencies of bi-grams f_{ij},
2. PK scaling parameter s,
3. Strength of rebalancing r.

Two different strategies are investigated that differ in the way these factors are determined.

Table 4.17: Parameter selection in the three setups.

Setup	On training data	On test data
No PK (Sec. 4.5)	Channel selection HMM training Word insertion penalty	none
Global PK	Channel selection HMM training Word insertion penalty PK scaling factor Rebalancing factor	none
Session PK	Channel selection HMM training	(PK matrix f_{ij})[a] PK scaling factor Word insertion penalty

[a]PK matrices are computed from the actual (bi-gram) frequencies f_{ij} within the corresponding test data sub-blocks.

In the first setting—named *Global PK*—training and testing phases are strictly separated, following the cross-session scheme as described in the previous section (cf. Figure 4.33). This represents a conventional BCI setting. Besides model training and determination of the optimal word insertion penalty, the choice of all parameters that are relevant for PK incorporation is also based on training data only. Since the only *a priori* information that holds true for both training and test dataset is the appearance of the fixed tapping sequence in the first sub-block (SB1), the corresponding bi-gram frequencies (i. e. $\mathbf{f}^{\mathrm{SB1}}$) are used for the PK term. The scaling and rebalancing parameters are determined using two-dimensional exhaustive search. Ultimately, decoding performances are assessed on a separate test dataset.

The second setting—referred to as *Session PK*—focuses on incorporation of context-specific information. By its arrangement in sub-blocks with differing tasks for the subjects, the experimental paradigm offers the opportunity to investigate this aspect. To do so, parameters related to PK incorporation are adjusted to the corresponding sub-blocks. In particular, this means that the applied bi-gram model is based on the actual frequencies of movement pairs in the sub-block of interest. In order to appropriately describe the differing conditions in the sub-blocks, strength of PK and word insertion penalty are optimized on the test dataset. In doing so, it is ensured that the available PK is used optimally with respect to the applied incorporation approach (i. e. bi-gram models) and thus, an upper-boundary of the achievable performance gain is assessed. To determine the optimal values for PK strength s and word insertion penalty p in this setting, both parameters (as well as the rebalancing factor r) are initialized with the results from the Global PK setting followed by alternating one-dimensional[23] exhaustive searches for the optimal s or p, respectively

[23]Instead of alternating one-dimensional searches, a 2D exhaustive search could also be used. While the 2D

Table 4.18: Optimal parameter values for word insertions penalty p, PK scaling s, and rebalancing factor r in the Global PK setting. All parameter are optimized based on training data only. One dimensional exhaustive search is used to determine p (cf. Figure 4.35). The tuple (s, r) is selected by means of 2-d grid search (Figure 4.39).

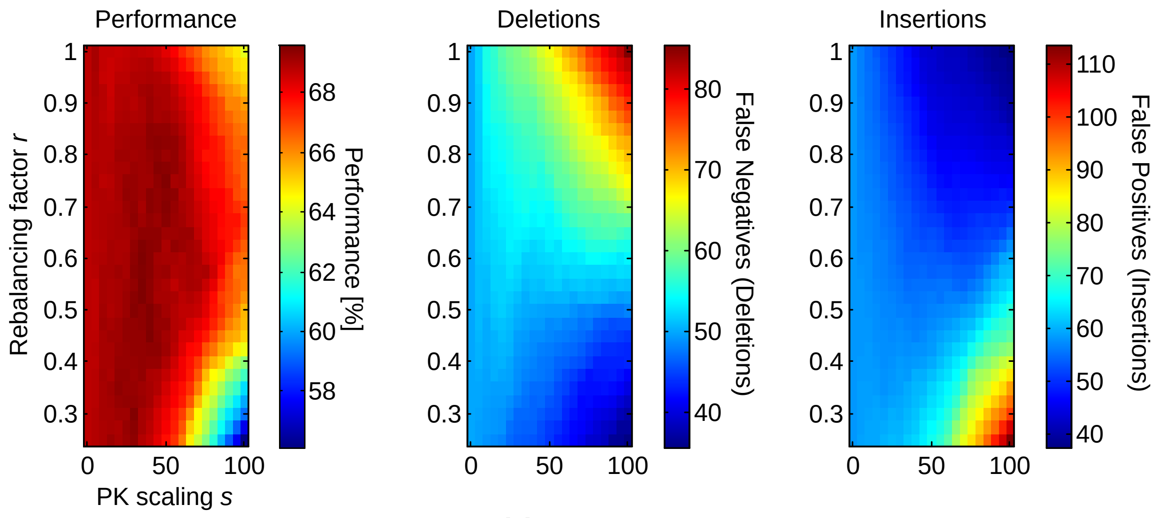
Dataset	p	s	r
S1-B1B2	56	40	0.45
S1-B2B1	77	25	0.85
S3-B1B2	66	20	0.35
S3-B2B1	56	30	0.40
S4-B1B2	44	20	0.55
S4-B2B1	41	20	0.73

(starting with s). Five iterations of the routine are performed; that means, each parameter is updated five times.

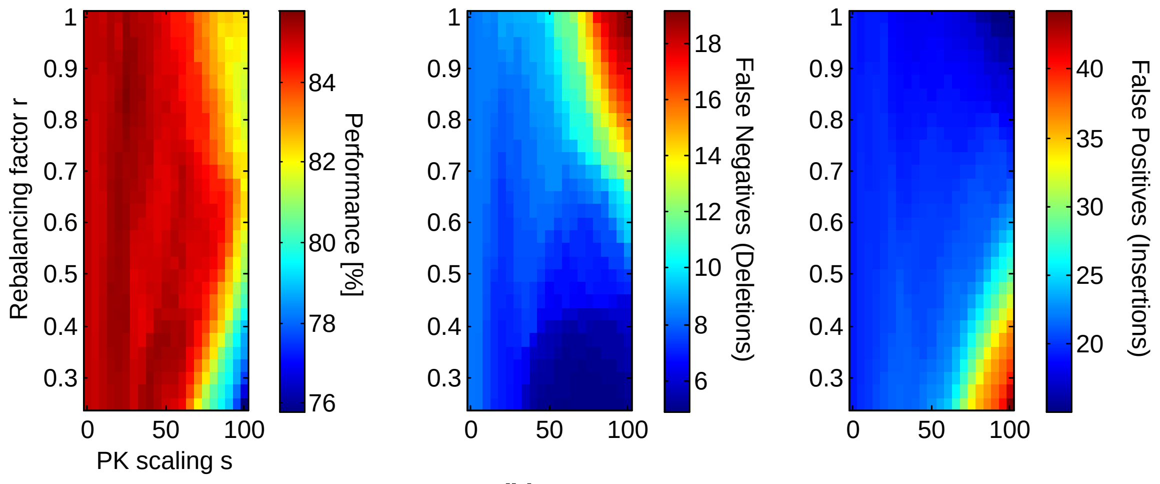
Table 4.17 summarizes the parameter selection settings for both setups. Additionally, the setting from the previous section (*No PK*, cf. Section 4.5) is listed for comparison.

Results For the Global PK setup, PK parameters are selected using two-dimensional exhaustive search (also called *grid search*) on the training dataset. Exemplary results of such a grid search are shown in Figure 4.39 for data from subject S1. The parameter tuple (s, r) that leads to the highest performance (left column in Figure 4.39) is selected for decoding of the test dataset. Table 4.18 shows the selected parameter for all datasets. In advance of 2D grid search for optimal s and r, word insertion penalty p has been optimized in a separate exhaustive search (as in Section 4.5). Note that word insertion penalties cannot be reused from the previous section (No PK) since Global PK is based on the modified dictionary containing combined "movement words". This leads to different optimal values for p. To provide detailed analysis of the influence of both components of PK (i. e. MRalt and Bi-Gram), decoding is performed once with $s = 0$ to assess the isolated performance gain of MRalt, and a second time with the actual optimized tuple of (s, r). The resulting prediction then contains the performance benefit of both PK components. Figure 4.40 (a) shows the decoding performances in the Global PK setting. Results from the No PK setting have been taken from Section 4.5 and are shown for comparison. The individual contributions of both components of PK (Figure 4.40 (b)) have been assessed with two

version would guarantee to find the global optimum and would appear conceptually simpler, it goes along with significantly higher computational effort. Keeping computation times at a manageable level has high priority since optimization needs to be done individually for each sub-block for all of the datasets. Hence, the alternating 1D approach has been used here.



(a) S1-B1B1



(b) S1-B2B2

Figure 4.39: Grid search results for PK parameters in the Global PK setting. Results are shown exemplary for subject S1 (training and decoding performed on the same dataset). Decoding performance with varying PK parameters are shown in the left column. The parameter tuple (s, r) that leads to the highest performance value is selected for decoding of the corresponding test dataset. Additionally, count of false negatives (center column) and false positives (right column) is shown. The influence of rebalancing factor r can be observed in these plots: High values of r imply higher penalties for additional detections and hence, help reduce the count of insertions. However, the amount of deletions raises at the same time. (Figure reproduced from [46].)

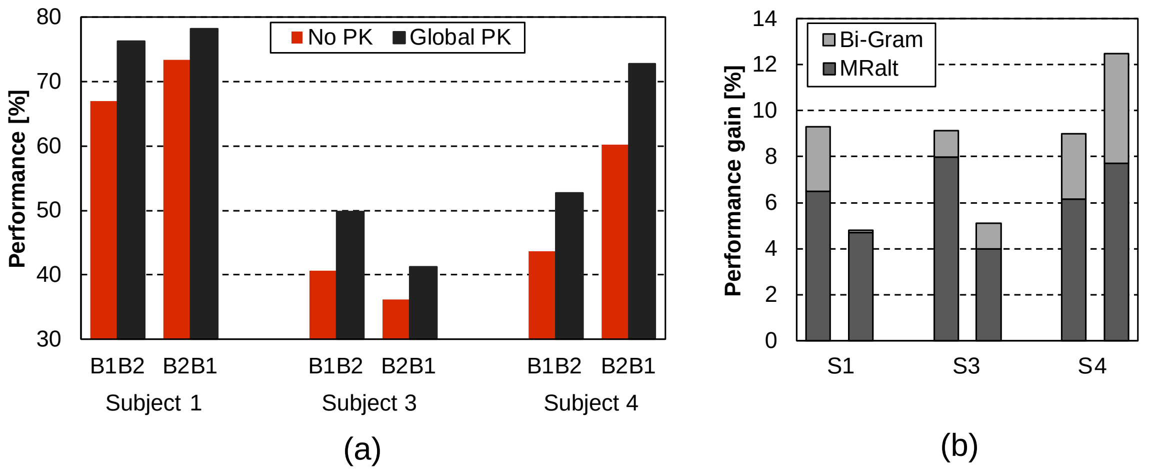


Figure 4.40: Performance comparison between decoding with and without PK incorporation. (a) Decoding performances in the No PK and Global PK settings for all subjects. (b) Individual contributions to the performance gain with Global PK caused by the Bi-Gram and MRalt component. (Figure reproduced from [46].)

separate decoding procedures as described above and computed as follows:

$$\Delta P^{\text{MRalt}} = P^{\text{Global PK}}(s=0,\cdot) - P^{\text{No PK}} \tag{4.5}$$

$$\begin{aligned} \text{and } \Delta P^{\text{Bi-Gram}} &= P^{\text{Global PK}}(s_{\text{opt}}, r_{\text{opt}}) - P^{\text{Global PK}}(s=0,\cdot) \qquad (4.6)\\ &= \left[P^{\text{Global PK}}(s_{\text{opt}}, r_{\text{opt}}) - P^{\text{No PK}}\right] - \Delta P^{\text{MRalt}} \\ &= \Delta P^{\text{Global PK}} - \Delta P^{\text{MRalt}}. \end{aligned}$$

It can be clearly seen that the MRalt component provides the dominant contribution to the overall performance gain of the Global PK setup. On average across all subjects and sessions, 8.3 % higher performance (absolute increase) is achieved with the Global PK setting. This performance gain is composed of 6.2 % resulting from the MRalt component and another 2.1 % of increase that are caused by the Bi-Gram component. The highest benefit is reported for S4-B2B1 with an overall performance gain of 12.5 % (7.7 % + 4.8 %). Across all datasets, the average PK entropy according to Eq. (3.4) is 3.99 ± 0.13 (cf. Table 4.16). In addition to the results for the full datasets, performances with the Global PK approach are also documented for individual sub-blocks in Table 4.19.

Session PK Introduction of generalized PK with the Global PK approach shows decent performance gain in the majority of cases. However, benefit of the Bi-Gram component is rather limited for most of the datasets. In the following, it will be investigated to what extend this benefit increases when more task-specific PK is considered. Since each sub-block in the experiment comprises different tasks, PK has been adapted to the specific conditions in the respective part of the data. First, the frequencies of all pairs of consecutive finger movements in the sub-block are computed and used within the bi-gram model that realizes

Table 4.19: Decoding performances P (in %) for individual sub-blocks in the No PK and Global PK setting. Additionally, individual performance gains (in %) from the MRalt component (ΔP^{MRalt}) and the Bi-Gram model ($\Delta P^{\text{BG}} := \Delta P^{\text{Bi-Gram}}$) in Global PK setting are documented. Benefit from the forced movement–rest alternation (with MRalt) is approximately the same for all sub-blocks. Contrary, the Bi-Gram component provides benefit mainly for SB1.

	Sub-block 1			Sub-block 2			Sub-block 3		
	No PK	Global PK		No PK	Global PK		No PK	Global PK	
Dataset	P	ΔP^{MRalt}	ΔP^{BG}	P	ΔP^{MRalt}	ΔP^{BG}	P	ΔP^{MRalt}	ΔP^{BG}
S1-B1B2	65.33	+7.67	+4.08	68.58	+6.17	+1.50	N/A	N/A	N/A
S1-B2B1	72.39	+7.11	+0.50	81.36	+3.56	- 0.17	71.05	+2.47	- 0.41
S3-B1B2	41.22	+2.24	+3.20	49.04	+10.96	- 0.11	15.17	+6.91	+0.94
S3-B2B1	31.58	+8.70	+3.96	52.52	+2.26	- 2.10	- 1.00	+8.44	- 5.79
S4-B1B2	45.38	+4.45	+4.59	41.84	+8.40	+2.10	45.32	+5.00	+1.53
S4-B2B1	63.17	+5.00	+9.91	62.38	+8.95	+2.67	48.00	+9.54	- 6.04
Ø	53.18	+5.86	+4.37	59.29	+6.72	+0.65	35.71	+5.39	- 1.63

Table 4.20: Final parameter values of word insertion penalty p, PK scaling s, and rebalancing factor r used in the Session PK setup.

		Session PK					
		Sub-block 1		Sub-block 2		Sub-block 3	
Dataset	r	p_{SB1}	s_{SB1}	p_{SB2}	s_{SB2}	p_{SB3}	s_{SB3}
S1 - B1	0.45	-93	46	-83	1	-25	18
S1 - B2	0.85	-40	42	-38.33	32	N/A	N/A
S3 - B1	0.35	-60	26	-49	31	-92	58
S3 - B2	0.40	-60.66	28.33	-66	66	-66	23
S4 - B1	0.55	-33	49	-37.66	18	-19	4
S4 - B2	0.73	-26.66	40	-28	62	-8	28

the corresponding PK component. Using the constructed bi-gram model, PK scaling s and word insertion penalty p are iteratively optimized on the test data (see 'Approach' paragraph). Table 4.20 shows the results of the optimization.

Decoding performances have been assessed individually for all sub-blocks of all datasets. Irrespective of the evaluated sub-block, decoder training has been performed on data of SB1 and SB2 from the other session (identical to the No PK and Global PK setup). Figure 4.41 shows the performance gain that is achieved with both components of incorporated PK in the Session PK setup for the different sub-blocks. To acquire the performance gain of the individual components (MRalt and Bi-Gram), a routine equivalent to the one described in the Global PK part has been applied (cf. Eq. (4.5) and (4.6)). The highest performance gains are reported for SB1 (avg. 14.1 %), which is expected since this sub-block contains the

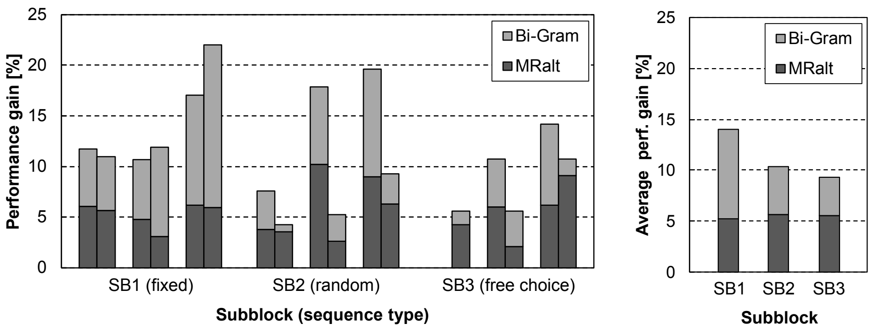


Figure 4.41: Performance gain with Session PK, broken down by individual contributions from the Bi-Gram and MRalt component. (left) Results for all sub-blocks from all datasets. The six bars for each sub-block correspond to the three subjects with two sessions each (i. e. S1-B1B2, S1-B2B1, ..., S4-B2B1). (right) Performance gain for the three sub-blocks averaged across all six datasets (three subjects, two sessions). (Left sub-figure reproduced from [46].)

largest amount of usable information due to the fixed tapping pattern. For SB2 and SB3, average performance gains of 10.4 % and 9.4 % are achieved, respectively. The difference between the three sub-blocks becomes even more apparent when looking at individual PK components. On average across all datasets, the benefit of MRalt is nearly identical for all sub-blocks (5.3 % – 5.7 %). This meets expectations, as the corresponding assumption of strict alternation between movement and rest is equally fulfilled in all sub-blocks. Contrary, the Bi-Gram component provides approximately twice the benefit for SB1 than for the other two sub-blocks (8.8 % vs. 4.7 % / 3.8 %).

Figure 4.42 shows the full decoding output of SB1 from S4-B2B1 decoding. This represents the scenario with highest performance gain (5.9 %+16.1 %=22.0 %). As seen in Figure 4.42, quite a large number of substitutions—especially between index and middle finger—are corrected by the introduction of Session PK. Besides that, it also appears that the count of deletions is reduced. Note that these results compare Session PK to No PK, and hence, represent combined benefit of both components of PK (MRalt and Bi-Gram). Since the influence of MRalt on the decoding accuracy has already been investigated in the previous section, performance gains that result from the Bi-Gram component shall be the focus of interest here. Predicted sequences, like the one shown in Figure 4.42, represent only a "snapshot" for a single run (with one particular seed for model initialization). To draw meaningful conclusions, it is important to compare the actual numbers of decoding errors for averaged (here: ten repetitions with differing seed) results (Table 4.21 and Appendix: Table A.4). Apparently, the observation from the snapshot that a considerable amount of substitutions is corrected ($\Delta S = -15.4$), holds true on averaged results. This observation also clarifies that the reduction of substitutions can be attributed to the Bi-Gram component. Compared to the substitutions, the decrease in insertions and deletions is markedly smaller ($\Delta I = -2.8$ and $\Delta D = -1.1$).

With more than 16.1 %, the resulting performance gain for SB1 is substantially higher than for the other two sub-blocks. This reflects that SB1 contains more usable PK. On average across all datasets, this observation is confirmed, albeit to a much lesser extend.

Table 4.21: Differences in count of matches and decoding errors (substitutions, insertions, and deletions) that result from applying the Bi-Gram component in Session PK. Results are shown for dataset S4-B2B1 (left) and on average across all datasets (right). Values that have negative impact on decoding performance are shown in red. Results for all other datasets are documented in the Appendix: Table A.4.

S4-B2B1	Sub-block		
	SB1	SB2	SB3
Matches	+16.5	+2.8	+1.1
Substitutions	- 15.4	- 4.6	- 0.9
Insertions	- 2.8	- 2.2	+0.1
Deletions	- 1.1	+1.3	- 0.2
Performance	**+16.1**	**+3.0**	**+1.6**

Ø All data	Sub-block		
	SB1	SB2	SB3
Matches	+7.6	+3.7	+4.7
Substitutions	- 7.8	- 4.5	- 0.6
Insertions	- 3.1	- 3.7	+1.9
Deletions	+0.2	+0.6	- 3.9
Performance	**+8.8**	**+4.7**	**+3.8**

Still, the main benefit originates from the reduction in substitutions and turns out to be highest for SB1.

An estimation of the information content that can be used for PK incorporation is given by the entropy measure defined by Eq. (3.4). This can be computed individually for each sub-block of all datasets (cf. Table 4.16). Subsequently, entropy values can be compared to the resulting performance gain that is achieved by incorporation of the corresponding PK. Such a comparison provides insight into whether there is a correlation between information content and performance gains. The entropy measure takes into account only the information that is utilized in the bi-gram model. Hence, isolated performance gains from the Bi-Gram component are considered here. Figure 4.43 shows the performance gain for all individual sub-blocks in dependence on the PK entropy. A general trend of increasing performance gain for lower entropy values (i. e. higher information content) is observed. Linear fitting results in a (squared) correlation coefficient $r^2 = 0.28$ under consideration of all data points. When leaving out the most questionable data point (SB3 from S4-B1B2), the fit ends up with $r^2 = 0.40$. The mentioned data point belongs to a dataset with by far the least information content (with respect to bi-gram entropy), but still provided relatively high performance increase (fourth highest among all data). It is likely that this performance gain is caused by influences that are uncorrelated to bi-gram information. Consequently, analysis has been carried out with and without the affected data point. Irrespective of whether or not the point is considered, linear fitting results in a positive slope for the relation

$$\Delta P(S) = aS + b$$

that is significant for a confidence level of 95 %. The resulting values for the slope and the corresponding confidence intervals are:

$$\begin{aligned} a &= 1.79\,(0.20, 3.38)_{\alpha=0.95} \\ a^* &= 2.29\,(0.67, 3.91)_{\alpha=0.95}, \end{aligned}$$

whereby a^* denotes the slope of the fit with the questionable data point left out. a^* is still significantly larger than 0 for a confidence level of 99 % (conf. bounds: $(0.04, 4.54)_{\alpha=0.99}$).

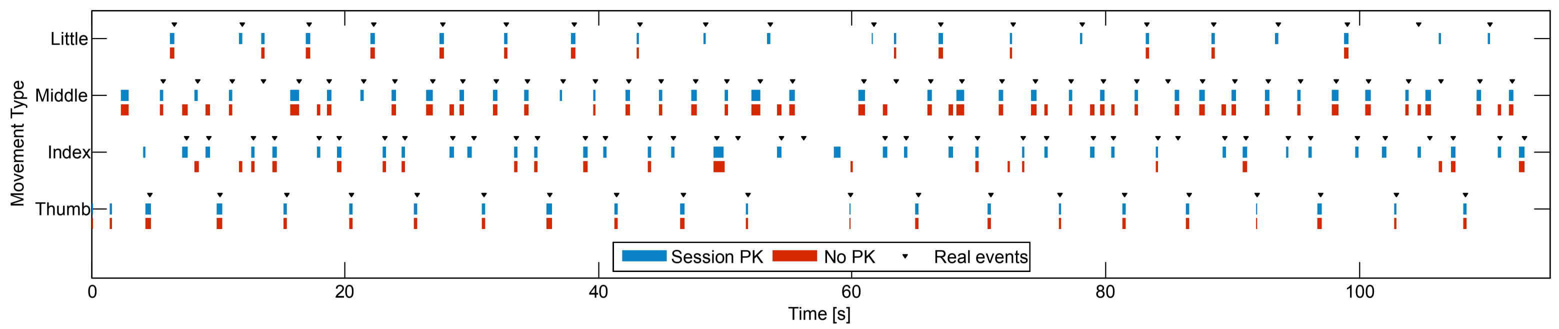


Figure 4.42: Continuous prediction output of the first sub-block (SB1) from S4-B2B1 decoding. Results from the Session PK (blue) setting are shown along with a prediction from No PK (red) for comparison. Actual button press events are indicated with black triangles. (Figure reproduced from [46].)

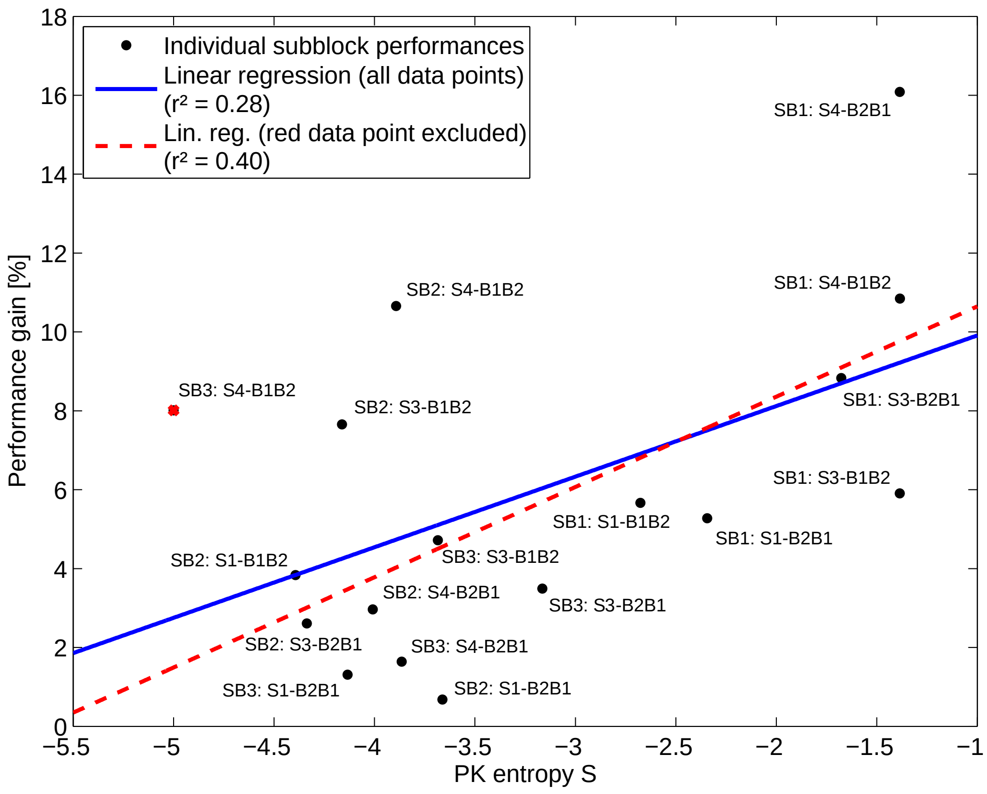


Figure 4.43: Performance gain of individual sub-blocks in the Session PK setup in dependence of their corresponding PK entropy. (Figure reproduced from [46].)

There is no theoretical justification on why the relation between PK entropy and performance gain should be linear. This specific functional relation has been chosen, as it represents the easiest realization of a monotonously increasing relation.

To facilitate comparison of the Session PK results to the findings from the other setups (No PK and Global PK), decoding results from the individual sub-blocks have been pooled to provide a total performance for the whole session. In practical terms, this means that performances have been averaged, weighted by the count of events in the corresponding sub-block:

$$\left\langle P^{\text{Session PK}} \right\rangle_{\text{w}} = \frac{1}{N} \sum_{i=1}^{3} N_{\text{SB}i} \cdot P_{\text{SB}i}^{\text{Session PK}} \qquad (4.7)$$

$$\text{with } N := N_{\text{SB1}} + N_{\text{SB2}} + N_{\text{SB3}}.$$

The resulting weighted averages are shown in Figure 4.44, along with the results from the No PK and Global PK setups. It becomes apparent that Session PK provides the highest accuracies throughout all datasets. On average, Session PK leads to a performance gain of

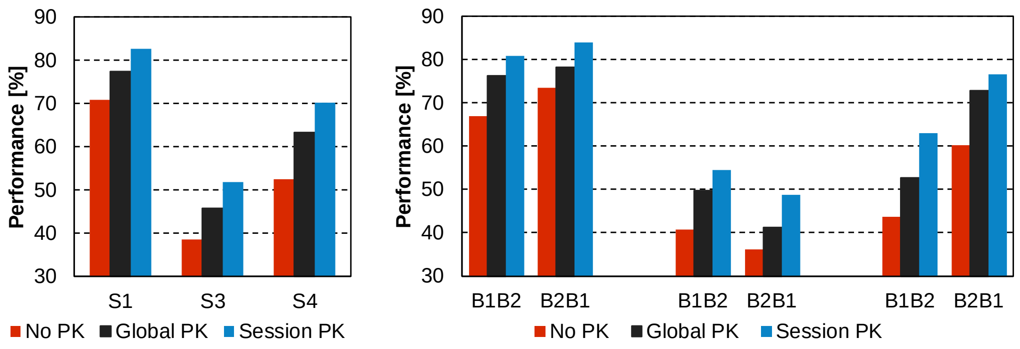


Figure 4.44: Comparison of decoding performances of all decoding settings. Performance values have been averaged across both sessions (left). Additionally, results for individual sessions are shown on the right. All results have been computed from the individual sub-block performances by means of weighted averaging (Eq. (4.7)). (Figure reproduced from [46].)

14.4 % (absolute increase) compared to decoding without PK incorporation (No PK). The performance increase with respect to Global PK is 6.0 % (average across all datasets).

Discussion With the Global PK setup, a generic setting for PK incorporation has been investigated. Enforcing strict alternation between movement and rest provides consistent increases in decoding accuracy across all datasets. This is expected, as the underlying assumption is fulfilled in all cases. In the Global PK configuration, the Bi-Gram component—which considers information on how frequent certain pairs of consecutive movements appear in the data—must also represent the entire dataset. As discussed before, a reasonable choice for that is to rely on the guaranteed occurrence of the fixed tapping pattern that is associated with SB1. However, as a consequence of the sub-structure in the experimental paradigm, any choice of generalized PK can only cover the properties of a fraction of the data. In the specific case investigated here, the Bi-Gram component meets the properties of between 30 % and 50 % of the total events in the data (depending on the specific ratio between SB1 and the remaining events, cf. Table 4.12). Hence, resulting performance gains are expected to be less substantial than for PK that holds true for the entire dataset. This is verified by the results, which expose that the benefit of the Bi-Gram component is approximately three times smaller than the MRalt performance gain. Still, for entire datasets, performance decreases cannot be observed in any of the investigated cases. This indicates that the routine is robust against introduction of potentially misleading information. The aspect has been examined more closely by analyzing the impact of the Bi-Gram component on individual sub-block level (Table 4.19). SB2 and SB3 are of particular interest in that regard, since the introduced PK does not fit for these sub-blocks. With the exception of two cases (SB3 of S3-B2B1 and S4-B2B1), performance changes are still positive or only slightly negative (-2.1 % ... +2.7 %). This shows that misleading information does not considerably corrupt predictions with the presented approach.

For the above mentioned reasons, the benefit of movement sequence consideration is limited for the Global PK approach. More detailed analysis was indicated to shed light on its true possibilities. In the Session PK setup, context-specific information is incorporated. In this setup, two parameters (PK incorporation strength s and word insertion penalty p) have been optimized based on the actual test data. Admittedly, this strategy cannot be used in real-life scenarios. However, essential insight into the capabilities the routine is granted, as the strategy provides an upper limit for the achievable performance gain. This can be interpreted as a setting in which best use (i. e. choosing ideal parameters) is made of the available PK, with respect to the chosen strategy to incorporate it into decoding (in this case: with bi-gram models). Although not transferable to real-life cases in straightforward manner, imaginable realistic scenarios could require similar concepts. When it comes to practical applications, patient-individual preferences create a demand for situation-dependent PK incorporation. This encompasses the possibility to switch the PK database itself—an opportunity that would, for example, allow the user to select from different "action protocols" like walking, eating/drinking or the like—but also to provide options to vary different aspects of decoder behavior. The latter could consist of adjusting (or fully enabling/disabling) the extent of PK incorporation or controlling the overall decoder sensitivity to adapt the BMI to higher/lower levels of activity. It is likely that such adaptions are requested by the user in a block-wise fashion (e. g. walking→drinking→walking→resting). The Session PK approach represents a model scenario for such a setting. More realistic realizations would involve parameter adaption in an on-the-fly manner (e. g. based on previous predictions); a concept sketch is presented in the outlook (Chapter 6, 'Improved PK incorporation'). It is unquestionable that those approaches would go along with slightly lower level of accuracy than the Session PK setup. Still, Session PK results approximate the possible benefit and can serve as a reference for the actually obtained outcome.

As expected, performance gain resulting from consideration of movement sequences (Bi-Gram) is substantially higher in the Session PK setup compared to the generalized approach in Global PK. In particular, decoding the first sub-block (SB1) benefits largely from incorporation of information on the fixed tapping sequence, which the patients performed during this part of the experiment. Depending on the subject, performances rise by 5.3 % to 16.1 %. Detailed analysis has revealed that the dominant effect of the Bi-Gram component is a reduction in substitutions. This matches expectations, since the bi-gram model introduces corrections to the probability of a certain prediction within a given context, in this case: the predecessor event. Such a correction can influence which model provides the highest total probability and hence, change the class that is predicted. Those adjustments are reflected by changes in the count of substitutions. One might be tempted to expect that SB1 can be decoded errorlessly when introducing PK, since the underlying tapping sequence is fully deterministic. However, this is not possible for several reasons: First of all, patients occasionally executed erroneous button presses. A single mistake immediately affects two transition frequencies: from the previous event to the button press ("the mistake") and from it to the next one. Thus, overall frequency matrix f_{ij} is altered substantially with each mistake. This becomes particularly apparent for subject S1, with an average entropy of $S = 2.51$ in SB1—for comparison, perfect SB1 execution would be $S^{\mathrm{SB1}} \approx 1.39$. Another important factor is the limited amount of available ground-truth information for the experimental data. Neither can it be guaranteed that button presses are conducted as a single clean movement, nor can it be precluded that movements of other fingers (not resulting in a button press), or at least muscular tension, happen at the same time. These aspects may relativize some of the decoding errors, since predicting a movement at a certain time point can actually match reality, although no event is documented for that particular moment (because no button has been pressed). Note that this can be attributed to the circumstance that it is not the button presses that are decoded but movements, induced by activation of muscles by their corresponding neural ensembles. In addition to imperfectly executed tapping sequences and the lack of detailed ground-truth data, there are some technical reasons that come into play. Bi-gram-based PK incorporation has no immediate influence on the timings of detections. To be more precise, it has no direct effect on the duration of rest intervals. Therefore, decoding errors that arise from wrong timing cannot be compensated for by that approach. Last, and most importantly, bi-gram models cannot reflect the entire information on the tapping sequence. As there are two possible successor events for both index finger and middle finger movements, at least a tri-gram model would be necessary to do so. Consequently, decoding either an index or middle finger movement leaves the decoder with two options for the possible next movement in each case. This degree of non-determinism is reflected by the entropy measure of the bi-gram model for SB1, which is non-zero; contrary to what it would be if it described a deterministic process in its entirety. Using higher order models to incorporate the PK for SB1 would likely lead to further increases in performance—probably up to near-perfect prediction. However, real life situations are unlikely to provide such encompassing PK, thus, rendering that setting unrealistic. For that reason, the use of bi-gram models can be seen as a well-suited compromise rather than a limitation here.

Although considerably smaller than for SB1, performance gains are also found for SB2 and SB3. At first glance, benefits from PK incorporation for SB2 might seem particularly surprising, given that the corresponding task is marked as 'random'. As stated in the paradigm description (Section 3.7.1), randomization has been restricted such that consecutive events cannot be of the same type. Therefore, the movement sequence still contains usable PK ($S^{\mathrm{SB2}} \approx 4.4 < 5.545 \approx S^{\max}(4)$). This amount of information is sufficient to compensate some of the decoding errors. Corrections mainly occur for substitutions and insertions. Reduction in substitutions is explained analogously to SB1. The elimination of some of the insertion errors results from the fact that erroneous fragmentation of detections into multiple individual events is prevented by the penalty that is applied when consecutive detections have the same type (Figure 4.45). In the last sub-block, subjects were granted the choice to use whatever finger they liked to perform the button press when indicated to do so. This caused rather high variance in the amount of usable PK across subjects (and sessions), since patient-individual movement preferences dominated the decision. Corresponding entropy values vary between 3.16 and 5.00 (Table 4.16) with an average of 3.97. This is a broad range, which averages just slightly below the randomized sequence from SB2. Consequently, performance gains differ strongly across the datasets. Similar to SB2, the available PK seems to be sufficient to correct a part of the decoding errors. Contrary to SB2, though, no meaningful conclusion can be drawn from the averaged changes in decoding errors (Table 4.21), since the averages are dominated by a single dataset (S3-B2B1,

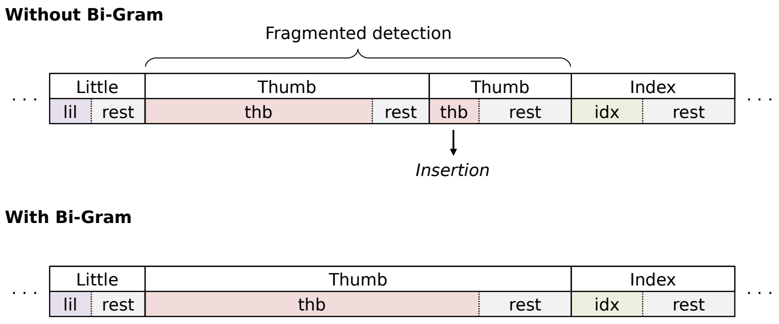


Figure 4.45: Illustration of detection fragmentation without use of the Bi-Gram component. In the given example, fragmentation can occur when a short intermediate data segment (here: just before the insertion) is better described by the rest model than the current movement model (here: thumb). This would result in a slightly higher overall likelihood for a fragmented sequence as shown on the top. Introducing a penalty for two consecutive detections of the same type (by means of bi-grams) can turn the likelihood in favor of a continuous segment (bottom).

Appendix: Table A.4). The performance for that particular dataset is rather low (23.9 %), thus, limiting its validity. Apart from that exception, most of the results are comparable to the other sub-blocks, suggesting that performance changes are induced by similar effects (i. e. reduced count of insertions etc.).

Technically, the constraint of the strict alternation between movement and rest has been realized by the construction of movement words. Instead, this could also be addressed by use of a bi-gram model that involves transition probabilities of 1.0 from any movement to the rest model, while transitions from rest to any of the movements are equally likely (0.25). The corresponding transition matrix is given by Eq. (4.8).

$$(f_{ij})^{\text{move-rest}} = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0.25 & 0.25 & 0.25 & 0.25 & 0 \end{pmatrix} \tag{4.8}$$

Although introducing the identical information as the combined word approach, a bi-gram-based strategy has a severe drawback in this context: When trying to simultaneously incorporate information on the frequencies of pairs of movements—as it is done with the Bi-Gram component—a simple bi-gram approach can no longer be used, since movement–rest alternation is already using up this option. Instead, at least a tri-gram model would be necessary to serve the purpose. Consequently, the alternative of using a dictionary of combined words has been preferred. However, interpretation of MRalt in terms of a possible bi-gram can provide interesting insight into the amount of information that is introduced by that component:

$$\begin{aligned} S(\mathbf{f}^{\text{move-rest}}) &= -\sum_{i=1}^{5}\sum_{j=1}^{5} f_{ij}^{\text{move-rest}} \cdot \ln f_{ij}^{\text{move-rest}} \\ &= -4 \cdot [1 \cdot \ln(1)] - 4 \cdot [0.25 \cdot \ln(0.25)] = 0 - 1 \cdot \ln(0.25) \\ &= -\ln(0.5^2) = 2\ln(2) \equiv S(\mathbf{f}^{\text{SB1}}) \approx 1.39. \end{aligned}$$

This means that the consideration of movement–rest alternation should provide the same amount of information as the movement sequence characteristics (Bi-Gram) from SB1 in the Session PK setup. Comparison of the results from the corresponding decoding procedures shows that this is fulfilled quite well for subjects S1 and S3 (avg. difference +1.5 %, in favor of Bi-Gram). However, for subject S4, the Bi-Gram component provides a substantially higher benefit (avg. difference +7.4 %) than MRalt.

Throughout this study, the entropy measure defined by Eq. (3.4) has been used to estimate the available information content for any given bi-gram model. The results show that this measure can serve as a rough approximation of information content. Qualitative correlation between entropy value and achievable performance gain is found on the level of averages across all subjects. However, on the level of individual subjects—and particularly for separate sub-blocks—the entropy measure cannot fully explain the different performance gains. For this purpose, the measure is too simplistic, considering the substantial level of complexity that is involved in continuous decoding of brain signals. The final influence of PK depends closely on very specific characteristics of decoding errors. While available PK may have high impact on decoder decisions between classes that are frequently mixed-up without the use of PK, it is certainly much less important for events that are already predicted reliably in a PK-free setting. Since HMMs are a probabilistic classifier and the presented PK incorporation approach manipulates those very probabilities, the effect of PK incorporation also depends on the probability differences between predicted (erroneous) and actual class. In a situation in which errors occur between two classes and there is only a minor probability difference between wrong and correct class, even the slightest amount of PK would possibly fix a large amount of errors. Furthermore, PK can only be beneficial, if (correctable) decoding errors occur in the first place. In case of near perfect decoding, performance gains are naturally much harder to achieve.

5. Discussion

5.1. Comparison to other studies

In general, comparison of results across different BCI studies is not a straightforward procedure. This is mainly due to the circumstance that almost every study is based on different recorded data—some studies might even use different recording modalities—resulting in significant differences in signal quality and information content. Furthermore, even in case of similar experimental tasks (e. g. finger tapping), the specific realization of the paradigm and thus, important practical conditions (e. g. length and intensity of the movements, use of contra- or ipsilateral hand) vary across those studies. With these limitations in mind, this section shall provide a comparison of methodology as well as (where possible) achieved results between this work and related studies.

Performance of HMM decoders Table 5.1 shows a selection of related studies that use HMM decoders in the context of BCI (cf. Table 1.1), including details on the used model and feature configuration. Additionally, a subjective rating is provided, relating the documented performance of the HMM routine to the approaches it has been compared with in those studies.

Some studies investigated the application of HMM-based approaches for the decoding of motor imagery, particularly for discriminating between left and right hand movement imagination. Especially the papers by Obermaier et al. [36], Sitaram et al. [37] and Lederman et al. [39] document substantial advantages when using HMM decoders. These studies use a left-to-right model topology in combination with a rather limited number of hidden states ($N \leq 5$) and Gaussian mixture components ($M \leq 5$). Interestingly, the amount of features used for decoding differs clearly ($n = 2...20$). However, due to the different recording modalities (EEG and NIRS), direct comparisons are difficult. Contrary to these positive results, two of the studies (Rezaei et al. [40] and Cincotti et al. [42]) report poor performance of HMMs. Unfortunately, both articles provide insufficient or almost no details on the concrete HMM configuration. Presumably, model topology has not been selected optimally in these studies, ending up in an unsuited setting for the investigated case. It is also noteworthy that those two articles represent comparative studies that investigate several different types of classifiers. Due to the rather extensive amount of adjustable parameters of HMM decoders (and their substantial influence on decoding outcome), selection of an appropriate configuration is a more delicate task than for many other routines.

The results presented in Section 4.1 clearly demonstrate that HMM-based decoding can provide similar decoding accuracies as an SVM approach (representing the class of less complex decoders). Consequently, the findings from Rezaei et al. and Cincotti et al. cannot

Table 5.1: Examples of BCI studies using HMM classification. N denotes the number of states in the HMM, M is the number of Gaussian mixture components in each state and n is the dimension of the feature vectors (i. e. number of features). Elements that are succeeded by a question mark are uncertain information that have been guessed from context, whereas a solitary question mark refers to information missing entirely.

Study	Model parameters				Features	HMM rating
	N	M	transition	n	type	
Obermaier [36]	1-5	1-5	LtR	2	Hjorth param.	++
Sitaram [37]	3	1/3/1	LtR	20	OxyHB/DeoxyHB	++
Lederman [39]	5	3	LtR	10	Filter bank coeff.	+/++
Chiappa [38]	2-7	3-15	full	19	PSD (4–40 Hz)	0
Cincotti* [42]	?	?	?	4?	12 spectral comp.	−−
Rezaei [40]	2	3	full?	8	AR coefficients	−−
Zhao [41]	2-6	?	full?	2	HG	N/A
Section 4.1	5	1	LtR (Bakis)	10/6–8	LFTD/HG	0
Section 4.3	4/5	1	LtR/±1	3+7	HG/LFTD	+

* very limited detail on used HMM configuration

Abbr.: AR–autoregressive model, HB–hemoglobin, LtR–left-to-right, PSD–power spectral density
Ratings (subjective): −− much worse, – worse, 0 equal, + better, ++ much better.

be confirmed. With a heavily constrained left-to-right topology using five states with a single mixture component, the applied model configuration in Section 4.1 is similar to the setup of other studies that document positive results for HMM decoders (e. g. Obermaier et al. [36], see above). Another important observation in the results from that section is that HMMs perform particularly well for high quality features (in this case: high gamma). Almost all other studies using HMM decoders (except for Zhao et al. [41]) are based on non-invasive recordings, which typically provide only mediocre signal quality. Consequently, those data do not allow for the use of high gamma features, and as a result, many studies used low frequency features. The results from Section 4.1 confirm that HMMs can turn out inferior in case of such features. Hence, underuse of HMM routines for decoding of invasively recorded data might be a reason for the overall low count in studies that report beneficial results of that approach.

As a general observation, it can be noted that many related studies feature long trial durations and are restricted to a small number of (mostly highly distinguishable) classes. Such conditions typically favor less complex classifiers. In addition, the mentioned studies focused entirely on decoding of class membership of trials in settings where all members of a class have identical properties[1] (e. g. all events have the same temporal extension). The full potential of dynamic classifiers remains unused in such setups. Chiappa and Bengio [38] investigated whether the dynamic aspect of HMM decoding can be beneficial

[1]This is also referred to as each class having only a single token.

by comparing its results to a configuration with only a single state—thus, creating a static setup. The authors used an asynchronous, imagery-based paradigm, which could potentially constitute a favorable case for dynamic classifiers. However, their findings suggest that no significant advantage of the dynamic approach can be found. As also discussed by the authors, this might be attributed to the use of fully connected transition matrices; these cannot consistently reproduce temporal characteristics.[2] The used model topology involved a huge parameter set with up to 7 states, 15 mixture components and 19 feature dimensions (yielding a total of 40,061 parameters, cf. Eq. (2.8)). By that, the authors used a rather high number of features (i. e. 19). As shown in Section 4.1, HMM decoders can be sensitive to an excessive amount of features, in particular when using low-frequency features (cf. Figure 4.4). In combination with the fully-connected topology, it is likely that such a model does not generalize well and hence, the static alternative—which naturally has much less parameters due to the low state count—provides better results.

With the investigations in Section 4.3, strong focus has been laid on demonstrating the potential of dynamic decoding. This is ensured by using a setting that involved increased complexity by means of considering additional attributes of the stimuli, which are now defined by both type and duration. The appearance of events with the same type but different temporal extent suggest that dynamic classifiers should be particularly suitable for decoding thereof. The results demonstrate that the properties of dynamic classification can be used to extract additional information about the event without dedicated training. An essential factor for these findings was an appropriate model topology. Contrary to Chiappa and Bengio [38], transition matrices have been restricted extensively and adapted to the specific requirements for the used feature types. Additionally, simplified model topology and problem-adapted feature selection with a reasonable number of selected features contributed substantially. This ensured representative, consistent state sequences (Viterbi paths), which in turn allowed for the extraction of event duration information with decent accuracy. As there is no direct analogue of such an approach for static classifiers, this verifies the superiority of dynamic classifiers in that context.

Finger tapping Decoding of finger movements is a rather common topic in BCI research. Consequently, several studies presented approaches and results in that regard. Table 5.2 shows an overview of studies that dealt with single trial finger movement decoding in a setting comparable to Section 4.1. With respect to the limitations discussed at the beginning of this section, comparison of the results shows that the HMM-based approach documented in this work competes well with other methods. This holds true particularly for the band power features (i. e. high gamma). The average decoding accuracy in this case is approximately equal to the best listed result (Scherer et al. [127]), and the maximum performance even clearly dominates the list. It further becomes apparent that for this rather demanding decoding task (four different finger movements of short duration), non-invasive data (Quandt et al. [121]) does not seem to provide a sufficiently high level of information to allow for precise decoding results.

[2]Note that this might be different if extensive training material would be available. However, this is typically not the case for BCI studies.

Table 5.2: Overview of studies reporting single trial finger tapping decoding results.

Study	Features	Mod.	Classifier	Dec. accuracy [%] Avg. (max.)
Liu [128]	spectral amplitudes	ECoG	SVM	83.8 (84.5)
Scherer [127]	band power	ECoG	LVQ	88.9 (91.3)
Shenoy [129]	band power	ECoG	SVM, LPM	77.0 (91.5*)
Quandt [121]	temporal (<16 Hz)	EEG	SVM	43.0 (54.5*)
		MEG		57.0 (70.0)
	spectrogram	MEG		36.0 (N/A)
Section 4.1	temporal	ECoG	HMM	74.2^{+} (80.7)
	band power			88.8^{+} (97.9)

LPM = linear programming machine, LVQ = learning vector quantization
* estimated from figures, $^{+}$ subject S2 excluded

Besides single trial studies (as discussed above), finger movements have also been decoded continuously (e. g. in [79, 130, 131]). However, the settings in these studies differ too substantially from the finger tapping experiment investigated here (cf. Section 3.7.1) to allow for meaningful comparison of the results.

Prior knowledge (PK) incorporation As mentioned in the very beginning, PK incorporation is found rather frequently within BCI spelling scenarios. These approaches typically use language models that are widely used in speech decoding and which are built from extensive databases. Using these models, substantial increases in system accuracy can be achieved [49]. This becomes possible as the desired application of spelling is related to speech and language. A closely related work is the study by Moses et al. [20] that deals with so-called neural speech recognition. In their work, the authors decode perceived speech directly from recorded ECoG signals in human subjects. They use linear discriminant analysis to estimate the likelihood of individual phonemes. A Viterbi decoder is applied to fuse these likelihoods with a phonemic language model that was built from a large database consisting of subtitles from many American films and television series. This decoding architecture is similar to a single state HMM routine. The incorporation of PK is done by means of n-gram models and the authors investigate the benefit for different level of the models (i. e. varying n). Significant improvements on decoding accuracy have been reported with that method compared to PK-free decoding.

However, for the case of motor BCI settings, comparable PK databases are unavailable. Hence, PK incorporation for those applications is a more complex task that requires problem-specific strategies. This is also reflected by the fact that no gold-standard approach for the purpose has established in the literature yet. Instead, several different methods have been used to consider other sources of information in order to increase the performance of BCI decoding in motor tasks. Most of these studies cannot be meaningfully compared to the methods presented here, due to fundamental differences in the data conditions.

In the method proposed by Delgado Saa et al. [53], conditional random fields (CRF) are used to decode finger flexion of individual fingers from ECoG signals. The authors introduce a so-called edge potential function as an additional (multiplicative) factor in the computation of their class probabilities. This would, in principle, allow them to consider arbitrary transition probabilities between different classes. However, the correction is used primarily to limit transition rates between different classes in order to account for the fact that no rapid changes between individual movements had been performed by the subjects.[3] Since rest episodes are considered in the same way as movements (as a sixth class), these restrictions also limit the change rate between movement and rest episodes. The limitation of transition rates with that approach has a comparable effect as word insertion penalties in the continuous HMM approach in Sections 4.5 and 4.6. With respect to the way it affects decoding of rest episodes, it is also distantly related to the movement–rest alternation (MRalt) method presented in Section 4.6. However, due to the algorithmic structure of the used edge potential function[4], only class transitions in directly adjacent time points are considered. The HMM-based approach on the contrary applies penalties for transitions between individual models. Since the decoder can dwell for several time samples in a single model, this describes rather long-time dependencies. The possibility of using their CRF method to incorporate higher-level information, such as frequencies of adjacent movements, is not discussed by the authors. Anyhow, the aforementioned structure of the edge potential would make it difficult to consider information similar to the Bi-Gram approach presented in Section 4.6 (as this refers to long-time dependencies). In fact, since the Bi-Gram method describes patterns consisting of triples: (movement 1, rest, movement 2), this could not be realized with that particular edge potential.

As far as movement decoding is concerned, there is no similar example in which the frequencies of subsequent individual events are used to provide the basis for PK incorporation. Additionally, no other documented study reported the use of n-grams to incorporate PK in motor tasks.

5.2. Limitations

Limited data pool All results presented in this work are based on rather small data pools. Particularly in case of electrocorticography (ECoG) recordings, databases are limited to 2–4 subjects. This has consequences for the generalizability of the findings. To increase statistic validity of the results, comparison of approaches should ideally be performed on larger databases. However, due to the difficulty of acquiring invasively recorded brain signals from larger patient collectives, small databases are very common in respective BCI studies.

[3]This might be explained by the fact that the underlying experimental paradigm consisted of fully-randomized movement sequences and hence, no other usable information except limited class change rates was available.

[4]The edge potential is a function $\Phi_i(y_i, y_{i-1})$ that depends on the labels y_i of the two adjacent time points i and $i-1$, with $i = 1, ..., n$ whereby n denotes the total number of time points in the data.

Invasive recordings The majority of findings presented here is based on invasively acquired data, which provide high signal quality. As shown in Sections 4.2 and 4.3, non-invasive data (in this case MEG) can also be decoded using the presented HMM decoders; however, the achievable level of accuracy is lower. It has been shown by Kübler et al. [132] that low decoding error rates are essential to prevent user frustration during use of a BCI in practice. Hence, invasive recordings are currently the most promising technique for meaningful implementation of complex BCI assistance systems. The major issue with recordings by means of ECoG or intracranial electrodes is their invasiveness. Both techniques go along with risk for the patient—especially with respect to possible infections due to the cable outlet—which is why the technique is still of limited use for real-life BCI solutions. However, current developments on the field of minimally invasive recording instrumentation (so-called stentrodes [133]) and wireless implant technology [134, 135] already show promising results, thus, potentially minimizing infection risk while maintaining the superior data quality.

Post-hoc analysis Throughout all parts of this work, analysis is based on *post-hoc* evaluation of pre-recorded data. While this is a conventional approach used in a large number of contemporary BCI studies, it has implications on the results that should be taken into consideration. An immediate consequence is that users do not receive any feedback on decoding results. Hence, decoding routines can only be adjusted to the recorded data (i. e. to the user), whereas training the user to the system is not feasible (closed-loop setup, "bi-directional training"). It is expectable that bi-directional training would further increase performances [136, 137], although potentially also requiring adaptions to the applied routines. In case of the single trial test procedure (i. e. by means of cross-validation), another aspect of the *post-hoc* evaluation comes into play. Training data within a CV routine contains trials from arbitrary time points within the dataset. If signal properties change throughout a recording session (e. g. due to the subject getting used to the experiment), this might change and blur the feature space representation [138]. In the CV routine, the classifier already "knows" about these upcoming changes, since the training set contains data from later time points. In an online setup, classifiers would not be trained to these changing characteristics. Consequently, decoding accuracies may be overestimated slightly compared to real online results. However, this holds true for any type of classifier and hence, does not unfairly benefit the HMM decoder in the results of Section 4.1.

Overt movements The investigated movement paradigm (finger tapping, see Section 3.7.1) is based on overt movements, i. e. patients actually performed the movements. Real-life BCI solutions are intended to assist paralyzed (or otherwise handicapped) patients with their every-day activities, making it impossible to rely on overt movements. Instead, decoding needs to be based on imagined or attempted movements. However, overt movements represent a well-suited and commonly used model for investigations of decoding approaches, primarily for two reasons: First, actually performed movements are easy to check, allowing for precise labeling of training data as well as meaningful analysis of evaluation datasets; both of which are difficult to accomplish when using attempted or imagined movements, as there is no conclusive proof of how well (and when precisely) the subject performed the imagination (or attempt). Second, imagined and particularly, attempted movements, go along with brain activity that is closely related to the one that appears in connection with actual movements [139, 140]. Recent studies emphasize that attempted movements might be the most reliable solution for real BCIs in paralyzed patients [141–143]. Since healthy subjects are unable to perform attempted movements[5], overt movements can serve as a suitable alternative.

Simultaneous events In this work, all decoding tasks consist of assigning a single label (from a set of discrete[6] values) to each sample (or entire trials) of the data. This is achieved using a maximum likelihood approach (Eq. (2.1)). While this strategy is fully sufficient in most applications (e. g. in speech decoding, as there can only be one articulation from a speaker at a time), it can imply limitations for specific demands (e. g. control of a hand prosthesis). In the example of finger movement decoding presented in Sections 4.5 and 4.6, prediction of multiple simultaneous movements would not be possible with the ML approach. With appropriate modifications to the decision criterion in the classification step, the option for multiple events could be realized. To do so, a threshold strategy on the likelihoods could be used. In that case, a movement would be predicted if the likelihood of the corresponding model exceeds a specified threshold. By that, several models could be "active" simultaneously. However, this would not only remove the ability to use well-established speech decoding tools straightforwardly, but also raises the demand for substantial modifications to the strategies of PK incorporation (away from simple n-gram models).

Parameter optimization A general limitation that applies to all parts of this work is the restriction to the investigation of isolated parameters influences. Analysis of the full variety of options at the same time is not feasible due to the enormous amount of parameters included in the applied routines. Exhaustive search—even on small parameter subsets—requires tremendously high computation times, making it impossible to practically perform such investigations. Besides these practical time issues, simultaneous variation of all available parameters also potentially results in locally optimal solutions for individual datasets ("overfitting"), which then cannot be transferred to other data.

5.3. Summary

It has been shown that HMM-based decoding can provide high accuracies in single trial BCI decoding. To achieve that, appropriate adaptions to the central components of the entire signal processing chain played an essential role. This includes finding optimal model configurations as well as suitable feature extraction and selection strategies. Ultimately, classification accuracies were brought to the level of gold-standard routines. In selected

[5]Unless by use of neuromuscular block, like in [142].

[6]Note that for duration decoding in Section 4.3, predicted durations were not restricted to a pre-defined set of values but could take (more or less) arbitrary values.

cases, HMMs outperformed the comparison method (i. e. support vector machines) with statistically significant performance advantages.

Furthermore, benefits of dynamic decoding have been demonstrated by the extraction of additional information about the decoded events. For this purpose, a method has been developed to analyze model output (or precisely, Viterbi paths), in order to assess the duration of an event without the requirement for a dedicated training routine—this is an approach without equivalent for static classifiers. In that context, it has also been shown that tailoring model configurations and adjusting topology restrictions to the specific properties of the used features, combined with problem-adapted channel selection strategies, are important contributions to all those findings.

Based on the aforementioned findings, the HMM routines have been extended to enable continuous decoding. In that context, it has been demonstrated how the gold-standard speech decoding framework HTK can be adapted for BCI decoding tasks. An extensive software framework—featuring a convenient graphical user interface for quick variation of all central decoding components—has been developed for this purpose. With two different approaches, it has been shown that HMM-based decoders offer the option for proper incorporation of prior knowledge from BCI scenarios. By effective utilization of language models, available *a priori* information could be exploited to substantially reduce decoding errors.

In summary, this work evidenced that the application of HMMs is a highly promising choice for decoding in the context of BCIs with great potential for practical application and further development. Some concepts for potential future investigations are presented in the closing chapter.

6. Outlook

As discussed in Section 5.2, the reported results underlie some limitations. While some of the mentioned restrictions are inherent to the problem, several other aspects might be addressed in future work. A selection of possible extensions and methodological refinements is presented here.

Continuous training This would provide the possibility to train models based on data for which one has no knowledge about the exact timings of the events. Instead of the exact time stamps of individual events, only the sequence in which events occur needs to be provided for training. With the HTK framework, this can be realized conveniently using the HERest subroutine. Training without requirement of timing of the events could be particularly valuable for imagery-based BCIs, as those imply difficulties to determine precisely when the subject starts imagining (and how long this continues).

Classifier adaptation With classifier adaption, model parameters could be re-adjusted on-the-fly during online use of the system. This has two immediate benefits. First, it increases the amount of training material for the classifier, as decoded 'test' data are reused for training; and second, adapting model parameters during use allows to compensate for changes in the brain signal characteristics (e. g. due to training effects of the subject) that might occur over time. However, to facilitate meaningful adaption strategies, precise interpretation of the decoded test data with respect to whether or not they have been decoded correctly is required. In order not to spoil model parameters with erroneous information, only events that are decoded correctly should be considered for online adaption. As there is no label information for test data, other solutions for this issue are required. Potential strategies could be decoding of so-called error potentials[1] or incorporation of some other sort of reliable user feedback (e. g. from other sources of information, like eye-tracking).

Closed-loop online setups The use of a closed-loop design would allow for 'bi-directional' training, i. e. both the decoder is trained to the brain responses of the user and the user itself is trained to modulate his/her thought processes by observing which responses they cause from the system. The latter component is not feasible with *post-hoc* data analysis (cf. Section 5.2).

More sophisticated paradigms The possible research questions that can be addressed in BCI studies are often limited by the specific characteristics of the available data. Experimental paradigms are usually tailored for very specific demands and hence,

[1]Error potentials are signal deflections that typically appear when the subject recognizes a mistake during a task. For a detailed description, see e. g. [144].

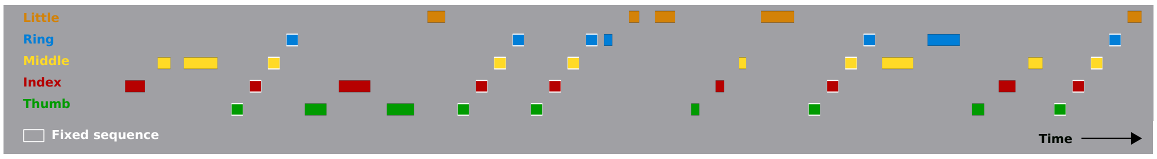


Figure 6.1: Exemplary snippet of a sequence for an augmented finger tapping paradigm.

are frequently not ideally suited for deviating requirements.[2] Particularly in case of invasively recorded data, new acquisition should not be carried out carelessly and hence, experiments would ideally be designed to allow for a broad spectrum of possible investigations.

A preliminary concept for a potential augmented finger tapping paradigm is briefly introduced in the following. Instead of just hitting a button on the keyboard for an arbitrarily short duration (as if it were pressed for reasons of typing), subjects would be requested to hold the button for a specified duration. The sequence of events would be constructed such that it, from time to time, contains certain blocks of a well-defined structure (i. e. with a fixed tapping pattern). These are interleaved by randomly generated events which could be drawn from a distribution that favors certain n-grams over others. An exemplary sequence is illustrated in Figure 6.1. Such a paradigm would allow for investigation of multiple aspects, separately as well as in arbitrary combination:

- continuous decoding with arbitrary event duration (combines Section 4.3 and 4.6),
- complex patterns of prior information (cf. outlook for 'Improved PK incorporation'),
- user training effects (recurrence of identical patterns),
- with further extensions: multiple simultaneous events (i. e. more than one finger active at a time).

Improved PK incorporation The strategy of incorporating PK could be extended to use of higher order models (i. e. up from bi-gram). This would allow considering more complex event sequences and thus, providing more flexibility for potential practical realizations. Additionally, to determine the PK rewards/penalties a straightforward approach is used that is simply based on frequencies of event sequences. More sophisticated strategies can be used to find reward/penalty values that are better suited. Instead of using frequencies of appearance as a template for PK penalties, one could use training data to estimate better values. To do so, one would classify some of the training data—if desired, this could be done in a nested cross-validation routine—and take a closer look at the occurring decoding errors. For the errors, the probability

[2]The picture category task, e. g., could not be used to study PK incorporation, due to the fact that stimulus type and duration had been selected entirely on a random basis. Therefore, no usable prior information on sequences is contained in these data.

differences between the predicted class and the correct class can be computed. Based on these differences, one could estimate penalty/reward values that would fix the corresponding decoding errors. Since any penalty or reward added to the decoding will affect all decoder decisions, it could result in new decoding errors. Hence, such a routine would likely need to be performed in an iterative way in order to determine values that lead to the best ratio of error corrections to newly introduced mistakes. This procedure could be advantageous, as it would allow for different strength of PK incorporation for specific class duels. This could become necessary when certain class combinations require only slight modification to fix decoding errors while others can only be corrected with larger interference.

Another possible extension to the methodology presented in Section 4.6 is closely related to classifier adaption (see above). To incorporate session-specific PK (Session PK setting), penalty/reward values have been determined using test data. Although legitimization of this strategy as a model scenario has been discussed in the corresponding result section, it is debatable to what extent the findings can be transferred to practical settings. A more realistic approach could be realized by starting with a PK free setting, decoding a certain amount of test data and using these data afterwards to refine the parameters of PK incorporation. Decoding of the subsequent segment of data could then benefit from improved PK parameters. The routine would be repeated each time a segment of data with a pre-defined length has been decoded—i. e. PK estimation can be performed on increasingly longer segments of data as decoding progresses. Contrary to the Session PK method from Section 4.6, this approach would only require "publishment" of the test data labels after decoding. While this still involves test data labels in some way, it could be interpreted as feedback from the user about the outcome of the latest performance of the system in a practical context.

PK incorporation has been limited to consider information that is inherent to the task.[3] Future work could also include fusion with other sources of information, like sensor data from prostheses, context information from cameras or various other sources of information.

Deep Learning Recent studies, especially in the field of automated speech recognition (ASR), investigated combinations of deep neural networks (DNN) and HMMs. Deep learning strategies can be used to provide optimized feature extraction or be included directly into the HMM architecture as a replacement for Gaussian mixture observation modeling (so-called DNN-HMMs [145]). Several approaches showed highly promising results in ASR [146,147], emotion recognition [148], and video action recognition [149]. These methods promise great potential for use in BCI contexts.

[3]This was necessary, since no other sources of information were available in the investigated scenario.

Bibliography

[1] E. C. Leuthardt, G. Schalk, D. W. Moran, and J. G. Ojemann. The emerging world of motor neuroprosthetics: A neurosurgical perspective. *Neurosurgery*, 59(1):1–14; discussion 1–14, 2006.

[2] J. R. Wolpaw, N. Birbaumer, W. J. Heetderks, D. J. McFarland, P. H. Peckham, G. Schalk, E. Donchin, L. A. Quatrano, C. J. Robinson, and T. M. Vaughan. Brain-computer interface technology: A review of the first international meeting. *Rehabilitation Engineering, IEEE Transactions on*, 8(2):164–173, 2000.

[3] H. Berger. Ueber das elektrenkephalogramm des menschen: I. mitteilung. *Archiv fuer Psychiatire*, 87(1):527–570, 1929.

[4] T. S. Moriyama, G. Polanczyk, A. Caye, T. Banaschewski, D. Brandeis, and L. A. Rohde. Evidence-based information on the clinical use of neurofeedback for adhd. *Neurotherapeutics*, 9(3):588–598, 2012.

[5] M. Mihara, N. Hattori, M. Hatakenaka, H. Yagura, T. Kawano, T. Hino, and I. Miyai. Near-infrared spectroscopy-mediated neurofeedback enhances efficacy of motor imagery-based training in poststroke victims: A pilot study. *Stroke*, 44(4):1091–1098, Feb 2013.

[6] B. I Morshed. A brief review of brain signal monitoring technologies for bci applications: Challenges and prospects. *Journal of Bioengineering & Biomedical Science*, 04(01), 2014.

[7] V. A. Maksimenko, S. van Heukelum, V. V. Makarov, J. Kelderhuis, A. Lüttjohann, A. A. Koronovskii, A. E. Hramov, and G. van Luijtelaar. Absence seizure control by a brain computer interface. *Scientific Reports*, 7(1), May 2017.

[8] D. Marshall, D. Coyle, S. Wilson, and M. Callaghan. Games, gameplay, and bci: The state of the art. *IEEE Transactions on Computational Intelligence and AI in Games*, 5(2):82–99, Jun 2013.

[9] B. Kerous, F. Skola, and F. Liarokapis. Eeg-based bci and video games: a progress report. *Virtual Reality*, pages 1–17, 2017.

[10] J. J. Vidal. Toward direct brain-computer communication. *Annu Rev Biophys Bioeng*, 2:157–180, 1973.

[11] N. Birbaumer, N. Ghanayim, T. Hinterberger, I. Iversen, B. Kotchoubey, A. Kübler, J. Perelmouter, E. Taub, and H. Flor. A spelling device for the paralysed. *Nature*, 398(6725):297–298, 1999.

[12] E. C. Leuthardt, G. Schalk, J. R. Wolpaw, J. G. Ojemann, and D. W. Moran. A brain-computer interface using electrocorticographic signals in humans. *Journal of neural engineering*, 1(2):63, 2004.

[13] K. J. Miller, L. B. Sorensen, J. G. Ojemann, and M. D. Nijs. Ecog observations of power-law scaling in the human cortex. *Potentials*, 1(12):4, 2007.

[14] K. J. Miller, P. Shenoy, M. denNijs, L. B. Sorensen, R. P. N. Rao, and J. G. Ojemann. Beyond the gamma band: The role of high-frequency features in movement classification. *Biomedical Engineering, IEEE Transactions on*, 55(5):1634–1637, 2008.

[15] K. J. Miller. *Characteristic changes in electrocorticographic power spectra of the human brain*. PhD thesis, University of Washington, 2008.

[16] K. J. Miller, S. P. Zanos, E. E. Fetz, M. denNijs, and J. G. Ojemann. Decoupling the cortical power spectrum reveals real-time representation of individual finger movements in humans. *Journal of Neuroscience*, 29(10):3132–3137, 2009.

[17] N. Liang and L. Bougrain. Decoding finger flexion using amplitude modulation from band-specific ecog. In *European Symposium on Artificial Neural Networks - ESANN 2009*, pages 467–472, Bruges, Belgium, 2009.

[18] J. Kubánek, K. J. Miller, J. G. Ojemann, J. R. Wolpaw, and G. Schalk. Decoding flexion of individual fingers using electrocorticographic signals in humans. *Journal of Neural Engineering*, 6(6):066001, Oct 2009.

[19] B. N. Pasley, S. V. David, N. Mesgarani, A. Flinker, S. A. Shamma, N. E. Crone, R. T. Knight, and E. F. Chang. Reconstructing speech from human auditory cortex. *PLoS Biol*, 10(1):e1001251, 2012.

[20] D. A. Moses, N. Mesgarani, M. K. Leonard, and E. F. Chang. Neural speech recognition: Continuous phoneme decoding using spatiotemporal representations of human cortical activity. *Journal of neural engineering*, 13:056004, 2016.

[21] 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*, 485:372–375, Mai 2012.

[22] T. Yanagisawa, M. Hirata, Y. Saitoh, H. Kishima, K. Matsushita, T. Goto, R. Fukuma, H. Yokoi, Y. Kamitani, and T. Yoshimine. Electrocorticographic control of a prosthetic arm in paralyzed patients. *Ann Neurol*, 71:353–361, 2012.

[23] J. L. Collinger, B. Wodlinger, J. E. Downey, W. Wang, E. C. Tyler-Kabara, D. J. Weber, A. J. McMorland, M. Velliste, M. L. Boninger, and A. B. Schwartz. High-performance neuroprosthetic control by an individual with tetraplegia. *The Lancet*, 381(9866):557–564, Feb 2013.

[24] N.-J. Huan and R. Palaniappan. Neural network classification of autoregressive features from electroencephalogram signals for brain-computer interface design. *Journal of neural engineering*, 1(3):142, 2004.

[25] M. K. Hazrati and E. A. Abbas. An online eeg-based brain-computer interface for controlling hand grasp using an adaptive probabilistic neural network. *Medical Engineering & Physics*, 32(7):730–739, 2010.

[26] T. Wu, B. Yang, and H. Sun. Eeg classification based on artificial neural network in brain computer interface. In K. Li, X. Li, S. Ma, and G. Irwin, editors, *Life System Modeling and Intelligent Computing*, volume 97 of *Communications in Computer and Information Science*, pages 154–162. Springer Berlin Heidelberg, 2010.

[27] O. Aydemir and T. Kayikcioglu. Decision tree structure based classification of 5eeg6 signals recorded during two dimensional cursor movement imagery. *Journal of Neuroscience Methods*, 229(0):68–75, 2014.

[28] F. Akram, M. H. S, and T.-S. Kim. An efficient word typing p300-bci system using a modified 5t9\ interface and random forest classifier. *Computers in Biology and Medicine*, 56(0):30–36, 2015.

[29] P. Shenoy, K. J. Miller, J. G. Ojemann, and R. P. N. Rao. Generalized features for electrocorticographic bcis. *IEEE Trans Biomed Eng*, 55(1):273–280, 2008.

[30] N. Jrad, M. Congedo, R. Phlypo, S. Rousseau, R. Flamary, F. Yger, and A. Rakotomamonjy. sw-svm: Sensor weighting support vector machines for eeg-based brain-computer interfaces. *Journal of neural engineering*, 8(5):056004, 2011.

[31] K. Kashihara. A brain-computer interface for potential nonverbal facial communication based on eeg signals related to specific emotions. *Frontiers in Neuroscience*, 8(244), 2014.

[32] F. Lotte, M. Congedo, A. Lecuyer, F. Lamarche, and B. Arnaldi. A review of classification algorithms for eeg-based brain-computer interfaces. *Journal of neural engineering*, 4(2):R1–R13, 2007.

[33] M. Gales and S. Young. The application of hidden markov models in speech recognition. *Found. Trends Signal Process.*, 1(3):195–304, 2007.

[34] J. Baker, L. Deng, J. Glass, S. Khudanpur, C.-H. Lee, N. Morgan, and D. O'Shaughnessy. Developments and directions in speech recognition and understanding, part 1 [dsp education]. *Signal Processing Magazine, IEEE*, 26(3):75–80, 2009.

[35] B. Obermaier, C. Guger, and G. Pfurtscheller. Hidden markov models used for the offline classification of eeg data. *Biomedizinische Technik Biomedical engineering*, 44(6):158–162, 1999.

[36] B. Obermaier, C. Guger, C. Neuper, and G. Pfurtscheller. Hidden markov models for online classification of single trial 5eeg\ data. *Pattern Recognition Letters*, 22(12):1299–1309, 2001.

[37] R. Sitaram, H. Zhang, C. Guan, M. Thulasidas, Y. Hoshi, A. Ishikawa, K. Shimizu, and N. Birbaumer. Temporal classification of multichannel near-infrared spectroscopy signals of motor imagery for developing a brain–computer interface. *NeuroImage*, 34(4):1416–1427, 2007.

[38] S. Chiappa and S. Bengio. Hmm and iohmm modeling of eeg rhythms for asynchronous bci systems. In *In European Symposium on Artificial Neural Networks, ESANN*, 2004.

[39] D. Lederman and J. Tabrikian. Classification of multichannel eeg patterns using parallel hidden markov models. *Med Biol Eng Comput*, 50:319–328, 2012.

[40] S. Rezaei, K. Tavakolian, A. M. Nasrabadi, and S. K. Setarehdan. Different classification techniques considering brain computer interface applications. *Journal of neural engineering*, 3(2):139, 2006.

[41] R. Zhao, G. Schalk, and Q. Ji. Coupled hidden markov model for electrocorticographic signal classification. In *ICPR*, pages 1858–1862. IEEE Computer Society, 2014.

[42] F. Cincotti, A. Scipione, A. Timperi, D. Mattia, A. Marciani, J. Millan, S. Salinari, L. Bianchi, and F. Bablioni. Comparison of different feature classifiers for brain computer interfaces. In *Neural Engineering, 2003. Conference Proceedings. First International IEEE EMBS Conference on*, pages 645–647. IEEE, 2003.

[43] B. Obermaier, C. Neuper, C. Guger, and G. Pfurtscheller. Information transfer rate in a five-classes brain-computer interface. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 9(3):283–288, 2001.

[44] T. Wissel, T. Pfeiffer, R. Frysch, R. T. Knight, E. F. Chang, H. Hinrichs, J. W. Rieger, and G. Rose. Hidden markov model and support vector machine based decoding of finger movements using electrocorticography. *Journal of neural engineering*, 10(5):056020, 2013.

[45] T. Pfeiffer, N. Heinze, R. Frysch, L. Y. Deouell, M. A. Schoenfeld, R. T. Knight, and G. Rose. Extracting duration information in a picture category decoding task using hidden markov models. *Journal of neural engineering*, 13(2):026010, 2016.

[46] T. Pfeiffer, R. T. Knight, and G. Rose. Hidden markov model based continuous decoding of finger movements with prior knowledge incorporation using bi-gram models. *Biomedical Physics & Engineering Express*, 4(2):025007, Jan 2018.

[47] J. N. Mak, Y. Arbel, J. W. Minett, L. M. McCane, B. Yuksel, D. Ryan, D. Thompson, L. Bianchi, and D. Erdogmus. Optimizing the p300-based brain–computer interface: current status, limitations and future directions. *Journal of Neural Engineering*, 8(2):025003, Mar 2011.

[48] F.-B. Vialatte, M. Maurice, J. Dauwels, and A. Cichocki. Steady-state visually evoked potentials: Focus on essential paradigms and future perspectives. *Progress in Neurobiology*, 90(4):418–438, Apr 2010.

[49] A. Mora-Cortes, N. V. Manyakov, N. Chumerin, and M. M. van Hulle. Language model applications to spelling with brain-computer interfaces. *Sensors*, 14(4):5967–5993, 2014.

[50] W. Speier, C. Arnold, and N. Pouratian. Integrating language models into classifiers for bci communication: A review. *Journal of neural engineering*, 13(3):031002, 2016.

[51] Z. Wang, Q. Ji, K. J. Miller, and G. Schalk. Prior knowledge improves decoding of finger flexion from electrocorticographic signals. *Frontiers in Neuroscience*, 5:127, 2011.

[52] R. Flamary and A. Rakotomamonjy. Decoding finger movements from ecog signals using switching linear models. *Frontiers in Neuroscience*, 6:29, 2012.

[53] J. F. D. Saa, A. d. Pesters, and M. Cetin. Asynchronous decoding of finger movements from ecog signals using longrange dependencies conditional random fields. *Journal of neural engineering*, 13:036017, 2016.

[54] G. Hotson, R. J. Smith, A. G. Rouse, M. H. Schieber, N. V. Thakor, and B. A. Wester. High precision neural decoding of complex movement trajectories using recursive bayesian estimation with dynamic movement primitives. *IEEE robotics and automation letters*, 1(2):676–683, 2016.

[55] M.-C. Schaeffer and T. Aksenova. Switching markov decoders for asynchronous trajectory reconstruction from ecog signals in monkeys for bci applications. *Journal of physiology, Paris*, 2017.

[56] T. Gasser and L. Molinari. The analysis of the eeg. *Statistical methods in medical research*, 5(1):67–99, 1996.

[57] A. F. Jackson and D. J. Bolger. The neurophysiological bases of eeg and eeg measurement: A review for the rest of us. *Psychophysiology*, 51(11):1061–1071, 2014.

[58] E. Niedermeyer and F. L. da Silva. *Electroencephalography: basic principles, clinical applications, and related fields*. Lippincott Williams & Wilkins, 2005.

[59] M. G. Stokes. Simple metric for scaling motor threshold based on scalp-cortex distance: Application to studies using transcranial magnetic stimulation. *Journal of Neurophysiology*, 94(6):4520–4527, Dec 2005.

[60] M. S. Beauchamp, M. R. Beurlot, E. Fava, A. R. Nath, N. A. Parikh, Z. S. Saad, H. Bortfeld, and J. S. Oghalai. The developmental trajectory of brain-scalp distance from birth through childhood: Implications for functional neuroimaging. *PLoS ONE*, 6(9):e24981, Sep 2011.

[61] D. Cohen. Magnetoencephalography: Detection of the brain's electrical activity with a superconducting magnetometer. *Science*, 175(4022):664–666, Feb 1972.

[62] M. W. Slutzky, L. R. Jordan, T. Krieg, M. Chen, D. J. Mogul, and L. E. Miller. Optimal spacing of surface electrode arrays for brain-machine interface applications. *Journal of neural engineering*, 7(2):26004, 2010.

[63] C. Babiloni, V. Pizzella, C. D. Gratta, A. Ferretti, and G. L. Romani. Chapter 5 fundamentals of electroencefalography, magnetoencefalography, and functional magnetic resonance imaging. *International Review of Neurobiology*, pages 67–80, 2009.

[64] R. Salmelin, M. Hámáaláinen, M. Kajola, and R. Hari. Functional segregation of movement-related rhythmic activity in the human brain. *NeuroImage*, 2(4):237–243, Dec 1995.

[65] M. Hämäläinen, R. Hari, R. J. Ilmoniemi, J. Knuutila, and O. V. Lounasmaa. Magnetoencephalography—theory, instrumentation, and applications to noninvasive studies of the working human brain. *Rev. Mod. Phys.*, 65(2):413–497, April 1993.

[66] F. Lopes da Silva. Functional localization of brain sources using eeg and/or meg data: volume conductor and source models. *Magnetic Resonance Imaging*, 22(10):1533–1538, Dec 2004.

[67] L. F. Nicolás-Alonso and J. G. Gil. Brain computer interfaces, a review. *Sensors*, 12(2):1211–1279, 2012.

[68] A. Bashashati, M. Fatourechi, R. K. Ward, and G. E. Birch. A survey of signal processing algorithms in brain–computer interfaces based on electrical brain signals. *Journal of Neural Engineering*, 4(2):R32–R57, Mar 2007.

[69] T. W. Picton, O. Lins, and M. Scherg. The recording and analysis of event-related potentials. *Handbook of Neuropsychology*, 10, 1995.

[70] A. Funase, T. Yagi, A. K. Barros, A. Cichocki, and I. Takumi. Single trial method for brain-computer interface. *2006 International Conference of the IEEE Engineering in Medicine and Biology Society*, Aug 2006.

[71] R. Q. Quiroga and H. Garcia. Single-trial event-related potentials with wavelet denoising. *Clin Neurophysiol*, 114(2):376–390, 2003.

[72] J. R. Wolpaw and D. J. McFarland. Control of a two-dimensional movement signal by a noninvasive brain-computer interface in humans. *Proceedings of the National Academy of Sciences of the United States of America*, 101(51):17849–17854, 2004.

[73] S. Makeig. Auditory event-related dynamics of the eeg spectrum and effects of exposure to tones. *Electroencephalography and Clinical Neurophysiology*, 86(4):283–293, Apr 1993.

[74] P. Clochon, J.-M. Fontbonne, N. Lebrun, and P. Etévenon. A new method for quantifying eeg event-related desynchronization: amplitude evvelope analysis. *Electroencephalography and Clinical Neurophysiology*, 98(2):126–129, Feb 1996.

[75] G. Pfurtscheller and F. H. L. d. Silva. Event-related eeg/meg synchronization and desynchronization: Basic principles. *Clinical Neurophysiology*, 110(11):1842–1857, 1999.

[76] A. Brovelli, J.-P. Lachaux, P. Kahane, and D. Boussaoud. High gamma frequency oscillatory activity dissociates attention from intention in the human premotor cortex. *NeuroImage*, 28(1):154–164, Oct 2005.

[77] R. T. Canolty, E. Edwards, S. S. Dalal, M. Soltani, S. S. Nagarajan, H. E. Kirsch, M. S. Berger, N. M. Barbaro, and R. T. Knight. High gamma power is phase-locked to theta oscillations in human neocortex. *Science*, 313(5793):1626–1628, Sep 2006.

[78] K. J. Miller, L. B. Sorensen, J. G. Ojemann, and M. den Nijs. Power-law scaling in the brain surface electric potential. *PLoS Computational Biology*, 5(12), 2009.

[79] Y. Nakanishi, T. Yanagisawa, D. Shin, C. Chen, H. Kambara, N. Yoshimura, R. Fukuma, H. Kishima, M. Hirata, and Y. Koike. Decoding fingertip trajectory from electrocorticographic signals in humans. *Neuroscience research*, 85:20–27, 2014.

[80] M. Yang, S. A. Sheth, C. A. Schevon, G. M. M. II, and N. Mesgarani. Speech reconstruction from human auditory cortex with deep neural networks. In *INTERSPEECH*, pages 1121–1125. ISCA, 2015.

[81] H. Song, D. Zhang, Z. Ling, H. cong Zuo, and B. Hong. High gamma oscillations enhance the subdural visual speller. In *EMBC*, pages 1711–1714. IEEE, 2012.

[82] Y. Akimoto, A. Kanno, T. Kambara, T. Nozawa, M. Sugiura, E. Okumura, and R. Kawashima. Spatiotemporal dynamics of high-gamma activities during a 3-stimulus visual oddball task. *PloS one*, 8(3):e59969, 2013.

[83] D. Zhang, H. Song, R. Xu, W. Zhou, Z. Ling, and B. Hong. Toward a minimally invasive brain–computer interface using a single subdural channel: a visual speller study. *Neuroimage*, 71:30–41, 2013.

[84] Z. J. Koles, M. S. Lazar, and S. Z. Zhou. Spatial patterns underlying population differences in the background eeg. *Brain Topography*, 2(4):275–284, 1990.

[85] M. Grosse-Wentrup and M. Buss. Multiclass common spatial patterns and information theoretic feature extraction. *IEEE Trans. Biomed. Engineering*, 55(8):1991–2000, 2008.

[86] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine Series 6*, 2(11):559–572, Nov 1901.

[87] R. A. FISHER. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, Sep 1936.

[88] A. Bhattacharyya. On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin of the Calcutta Mathematical Society*, 35:99–109, 1943.

[89] D. L. Davies and D. W. Bouldin. A cluster separation measure. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-1(2):224–227, 1979.

[90] S. H. Fairclough and K. Gilleade, editors. *Advances in Physiological Computing.* Human-Computer Interaction Series. Springer London, London and s.l., 2014.

[91] L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *Annals of Mathematical Statistics*, 37:1554–1563, 1966.

[92] A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, IT-13(2):260–269, April 1967.

[93] L. R. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition.* Englewood Cliffs, NJ: Prentice Hall, 1993.

[94] X. Huang, A. Acero, H.-W. Hon, and R. Reddy. *Spoken language processing: A guide to theory, algorithm, and system development*, volume 95. Prentice hall PTR Upper Saddle River, 2001.

[95] G. A. Fink. *Markov models for pattern recognition: from theory to applications.* Springer Science & Business Media, 2014.

[96] B.-H. Juang and L. R. Rabiner. The segmental k-means algorithm for estimating parameters of hidden markov models. *IEEE Trans. Acoustics, Speech, and Signal Processing*, 38(9):1639–1641, 1990.

[97] Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden markov support vector machines. *Proceedings of the Twentieth International Conference on Machine Learning*, 2003.

[98] L. R. Rabiner and B. H. Juang. Statistical methods for the recognition and understanding of speech. In *Encyclopedia of Language and Linguistics.* Elsevier, 2004.

[99] B. H. Juang and L. R. Rabiner. Automatic speech recognition - a brief history of the technology. In 2nd, editor, *Elsevier Encyclopaedia of Language and Linguistics.* Elsevier, 2005.

[100] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[101] C. Kurian. A review on technological development of automatic speech recognition. *International Journal of Soft Computing and Engineering (IJSCE)*, 4(4):80–86, 2014.

[102] A. Kosmala, J. Rottland, and G. Rigoll. *Large Vocabulary On-Line Handwriting Recognition with Context Dependent Hidden Markov Models.* Springer, 1997.

[103] R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids.* Cambridge University Press, 1998.

[104] A. Krogh. An introduction to hidden markov models for biological sequences. *New Comprehensive Biochemistry*, pages 45–63, 1998.

[105] T. Dickhaus, C. Sannelli, K.-R. Müller, G. Curio, and B. Blankertz. Predicting bci performance to study bci illiteracy. *BMC Neuroscience*, 10(1):1–2, 2009.

[106] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, (2):121–167, 1998.

[107] V. N. Vapnik. *Statistical Learning Theory.* Wiley-Interscience, September 1998.

[108] N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68:337–404, 1950.

[109] K.-R. Müller, S. Mika, G. Rätsch, S. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–202, 2001.

[110] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[111] S. J. Young. The htk hidden markov model toolkit: Design and philosophy. *Entropic Cambridge Research Laboratory, Ltd*, 2:2–44, 1994.

[112] R. Frysch. Vergleich und optimierung von klassifikatoren zur erkennung von fingerbewegungen auf der grundlage von direkt am gehirn gemessenen elektrischen potentialen. Diploma thesis, Otto-von-Guericke-Universitaet Magdeburg, 2012.

[113] T. Pfeiffer. Optimierung von merkmalen für die klassifikation direkt am kortex gemessener elektrischer potentiale für gehirn-computer-schnittstellen. Diploma thesis, Otto-von-Guericke-Universitaet Magdeburg, 2012.

[114] T. Pfeiffer, R. T. Knight, and G. Rose. Word networks for bci decoding purposes. In *Proceedings of the Sixth International Brain-Computer Interface Meeting: BCI Past, Present, and Future*, page 160, 2016.

[115] K. Murphy. Hidden markov model (hmm) toolbox for matlab, 2005.

[116] C.-C. Chang and C.-J. Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27, 2011.

[117] G. Pfurtscheller, B. Graimann, J. E. Huggins, S. P. Levine, and L. A. Schuh. Spatiotemporal patterns of beta desynchronization and gamma synchronization in corticographic data during self-paced movement. *Clinical Neurophysiology*, 114(7):1226–1236, 2003.

[118] K. J. Miller, E. C. Leuthardt, G. Schalk, R. P. N. Rao, N. R. Anderson, D. W. Moran, J. W. Miller, and J. G. Ojemann. Spectral changes in cortical surface potentials during motor movement. *Journal of Neuroscience*, 27(9):2424–2432, 2007.

[119] E. C. Leuthardt, Z. Freudenberg, D. Bundy, and J. Roland. Microscale recording from human motor cortex: Implications for minimally invasive electrocorticographic brain-computer interfaces. *Neurosurgical Focus*, 27(1):E10, 2009.

[120] T. Pfeiffer, N. Heinze, E. M. Gerber, L. Y. Deouell, J. Parvizi, R. T. Knight, and G. Rose. Decoding of picture category and presentation duration - preliminary results of a combined ecog and meg study. In *Proceedings of the 6th International Brain-Computer Interface Conference 2014*, pages ID 042–1, 2014.

[121] F. Quandt, C. Reichert, H. Hinrichs, H. J. Heinze, R. T. Knight, and J. W. Rieger. Single trial discrimination of individual finger movements on one hand: A combined meg and eeg study. *NeuroImage*, 59(4):3316–3324, 2012.

[122] K. Grill-Spector. The neural basis of object perception. *Current Opinion in Neurobiology*, 13(2):159–166, Apr 2003.

[123] J. V. Haxby, M. I. Gobbini, M. L. Furey, A. Ishai, J. L. Schouten, and P. Pietrini. Distributed and overlapping representations of faces and objects in ventral temporal cortex. *Science*, 293(5539):2425–2430, 2001.

[124] L. Fisch, E. Privman, M. Ramot, M. Harel, Y. Nir, S. Kipervasser, F. Andelman, Y. N. M, U. Kramer, I. Fried, and R. Malach. Neural ignition: Enhanced activation linked to perceptual awareness in human ventral stream visual cortex. *Neuron*, 64(4):562–574, 2009.

[125] L. Reddy, N. Tsuchiya, and T. Serre. Reading the mind's eye: Decoding category information during mental imagery. *NeuroImage*, 50(2):818–825, 2010.

[126] S. J. Young, D. Kershaw, J. Odell, D. Ollason, V. Valtchev, and P. Woodland. *The HTK Book Version 3.4*. Cambridge University Press, 2006.

[127] R. Scherer, S. P. Zanos, K. J. Miller, R. P. N. Rao, and J. G. Ojemann. Classification of contralateral and ipsilateral finger movements for electrocorticographic brain-computer interfaces. *Neurosurg Focus*, 26 (6), 2009.

[128] Y. Liu, M. Sharma, C. M. Gaona, J. D. Breshears, J. Roland, Z. V. Freudenburg, K. Q. Weinberger, and E. C. Leuthardt. Decoding ipsilateral finger movements from ecog signals in humans. *Advances in Neural Information Processing Systems*, 23:1468–1476, 2010.

[129] P. Shenoy, K. J. Miller, J. G. Ojemann, and R. P. N. Rao. Finger movement classification for an electrocorticographic bci. *Neural Engineering, 2007. CNE '07. 3rd International IEEE/EMBS Conference on*, pages 192–195, 2007.

[130] N. Liang and L. Bougrain. Decoding finger flexion from band-specific ecog signals in humans. *Frontiers in Neuroscience*, 6:91, 2012.

[131] W. Chen, X. Liu, and B. Litt. Logistic-weighted regression improves decoding of finger flexion from electrocorticographic signals. *Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual Conference*, 2014:2629–2632, 2014.

[132] A. Kübler, E. M. Holz, A. Riccio, C. Zickler, T. Kaufmann, S. C. Kleih, P. Staiger-Sälzer, L. Desideri, E.-J. Hoogerwerf, and D. Mattia. The user-centered design as novel perspective for evaluating the usability of bci-controlled applications. *PLoS ONE*, 9(12):e112392, 2014.

[133] T. J. Oxley, N. L. Opie, S. E. John, G. S. Rind, S. M. Ronayne, T. L. Wheeler, J. W. Judy, A. J. McDonald, A. Dornom, T. J. H. Lovell, and et al. Minimally invasive endovascular stent-electrode array for high-fidelity, chronic recordings of cortical neural activity. *Nature Biotechnology*, 34(3):320–327, Feb 2016.

[134] F. H. Guenther, J. S. Brumberg, E. J. Wright, A. Nieto-Castanon, J. A. Tourville, M. Panko, R. Law, S. A. Siebert, J. L. Bartels, D. S. Andreasen, P. Ehirim, H. Mao, and P. R. Kennedy. A wireless brain-machine interface for real-time speech synthesis. *PLoS ONE*, 4(12):e8218, 2009.

[135] D. A. Borton, M. Yin, J. Aceros, and A. Nurmikko. An implantable wireless neural interface for recording cortical circuit dynamics in moving primates. *Journal of Neural Engineering*, 10(2):026010, 2013.

[136] J. M. Carmena, M. A. Lebedev, R. E. Crist, J. E. O'Doherty, D. M. Santucci, D. F. Dimitrov, P. G. Patil, C. S. Henriquez, and M. A. Nicolelis. Learning to control a brain–machine interface for reaching and grasping by primates. *PLoS biology*, 1(2):e42, 2003.

[137] R. Héliot, K. Ganguly, J. Jimenez, and J. M. Carmena. Learning in closed-loop brain-machine interfaces: Modeling and experimental validation. *IEEE Trans. Systems, Man, and Cybernetics, Part B*, 40(5):1387–1397, 2010.

[138] S. Lemm, B. Blankertz, T. Dickhaus, and K.-R. Müller. Introduction to machine learning for brain imaging. *NeuroImage*, 56(2):387–399, 2011.

[139] H. Sugata, M. Hirata, T. Yanagisawa, K. Matsushita, S. Yorifuji, and T. Yoshimine. Common neural correlates of real and imagined movements contributing to the performance of brain–machine interfaces. *Scientific Reports*, 6(1), Apr 2016.

[140] A. M. Batula, J. Mark, Y. E. Kim, and H. Ayaz. Comparison of brain activation during motor imagery and motor movement using fnirs. *Comp. Int. and Neurosc.*, 2017:5491296:1–5491296:12, 2017.

[141] Y. Blokland, R. Vlek, B. Karaman, F. Ozin, D. Thijssen, T. Eijsvogels, W. Colier, M. Floor-Westerdijk, J. Bruhn, and J. Farquhar. Detection of event-related desynchronization during attempted and imagined movements in tetraplegics for brain switch control. *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Aug 2012.

[142] Y. Blokland, L. Spyrou, J. Lerou, J. Mourisse, G. Jan Scheffer, G.-J. v. Geffen, J. Farquhar, and J. Bruhn. Detection of attempted movement from the eeg during neuromuscular block: proof of principle study in awake volunteers. *Scientific Reports*, 5(1), Aug 2015.

[143] G. Müller-Putz, A. Schwarz, J. Pereira, and P. Ofner. From classic motor imagery to complex movement intention decoding. *Brain-Computer Interfaces: Lab Experiments to Real-World Applications*, pages 39–70, 2016.

[144] M. Falkenstein, J. Hoormann, S. Christ, and J. Hohnsbein. Erp components on reaction errors and their functional significance: a tutorial. *Biological Psychology*, 51(2-3):87–107, Jan 2000.

[145] D. Yu and L. Deng. *Deep Neural Network-Hidden Markov Model Hybrid Systems*. Springer, 2015.

[146] G. E. Dahl, D. Yu, L. Deng, and A. Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Trans. Audio, Speech & Language Processing*, 20(1):30–42, 2012.

[147] Z.-J. Yan, Q. Huo, and J. Xu. A scalable approach to using dnn-derived features in gmm-hmm based acoustic modeling for lvcsr. In *Interspeech*, pages 104–108, 2013.

[148] L. Li, Y. Zhao, D. Jiang, Y. Zhang, F. Wang, I. Gonzalez, E. Valentin, and H. Sahli. Hybrid deep neural network–hidden markov model (dnn-hmm) based speech emotion recognition. In *Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on*, pages 312–317. IEEE, 2013.

[149] J. Lei, G. Li, J. Zhang, Q. Guo, and D. Tu. Continuous action segmentation and recognition using hybrid convolutional neural network-hidden markov model model. *IET Computer Vision*, 10(6):537–544, 2016.

A. Appendix

A.1. Examples

Example from Section 3.6.3 Let us assume a three-class sequence with the following transitions:

$$
\begin{aligned}
11 &\to 3\\
13 &\to 2\\
32 &\to 1\\
21 &\to 2\\
12 &\to 2\\
22 &\to 3\\
23 &\to 3\\
33 &\to 1\\
31 &\to 1
\end{aligned}
$$

or in sorted order:

$$
\begin{array}{lll}
11 \to 3 & 21 \to 2 & 31 \to 1\\
12 \to 2 & 22 \to 3 & 32 \to 1\\
13 \to 2 & 23 \to 3 & 33 \to 1
\end{array}
$$

This leads to the following tri-gram transition frequencies (f_{ijk}):

$$
(f_{1jk}) = \begin{pmatrix} 0 & 0 & 1\\ 0 & 1 & 0\\ 0 & 1 & 0 \end{pmatrix}, \quad (f_{2jk}) = \begin{pmatrix} 0 & 1 & 0\\ 0 & 0 & 1\\ 0 & 0 & 1 \end{pmatrix}, \quad (f_{3jk}) = \begin{pmatrix} 1 & 0 & 0\\ 1 & 0 & 0\\ 1 & 0 & 0 \end{pmatrix}.
$$

Apparently, this sequence is fully deterministic. However, its corresponding bi-gram frequencies (f_{ij}) are

$$
(f_{ij}) = \begin{pmatrix} 1/3 & 1/3 & 1/3\\ 1/3 & 1/3 & 1/3\\ 1/3 & 1/3 & 1/3 \end{pmatrix}.
$$

Consequently, this sequence has PK entropy $S = S^{\max}(3)$. Information only on the current event does not allow for any prediction on the upcoming event.

Examples of visual stimuli

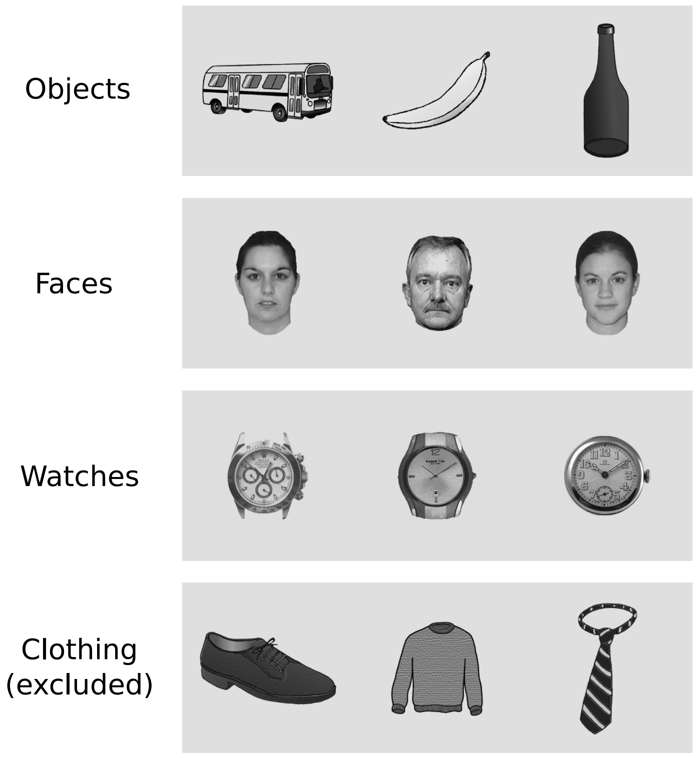


Figure A.1: Examples of visual stimuli used in the picture category experiment. Note that 'clothing' stimuli have been used only to keep up subjects' attention (ECoG experiment only) and have been excluded for analysis.

Example from Section 4.4 Let us assume that the first third ($\mathbf{x}_1$) of a given trial $\tilde{\mathbf{x}} = (\mathbf{x}_1, \mathbf{x}_2)$ represents the original stimulus A and the remaining two thirds ($\mathbf{x}_2$) contain an additional stimulus B.[1] In this example, the first segment shall be modeled with high likelihood by its corresponding HMM-A, e. g. $P_{\text{HMM-A}}(\mathbf{x}_1) = 0.8$ while the other HMM-B provides a significantly lower probability of $P_{\text{HMM-B}}(\mathbf{x}_1) = 0.1$. The remaining part is slightly in favor of HMM-B with $P_{\text{HMM-A}}(\mathbf{x}_2) = 0.5$ and $P_{\text{HMM-B}}(\mathbf{x}_2) = 0.6$. The resulting total probability would turn out clearly in favor of class A with the Vanilla HMM approach:

$$P_{\text{HMM-A}}(\tilde{\mathbf{x}}) = P_{\text{HMM-A}}(\mathbf{x}_1) \cdot P_{\text{HMM-A}}(\mathbf{x}_2) = \mathbf{0.40}$$
$$P_{\text{HMM-B}}(\tilde{\mathbf{x}}) = P_{\text{HMM-B}}(\mathbf{x}_1) \cdot P_{\text{HMM-B}}(\mathbf{x}_2) = 0.06$$

The same example would yield a different result with the semi-continuous approach. In that case, the first third of $\tilde{\mathbf{x}}$ would be modeled by HMM-A followed by a transit to model HMM-B, which models the rest of the trial (since it has a higher likelihood for that segment). The total probability would compute as follows:

$$P_{\text{WN default}}(\tilde{\mathbf{x}}) = P_{\text{HMM-A}}(\mathbf{x}_1) \cdot P^{\text{penalty}}_{\text{A}\to\text{B}} \cdot P_{\text{HMM-B}}(\mathbf{x}_2) = 0.48 \cdot P^{\text{penalty}}_{\text{A}\to\text{B}}$$

[1]Note that pauses are omitted for means of simplicity.

with $P_{\mathrm{A}\to\mathrm{B}}^{\mathrm{penalty}}$ denoting the penalty that is applied for each model transition from HMM-A to HMM-B. Such a penalty is introduced to prevent the decoder from overabundant model transitions. For penalty values of $P_{\mathrm{A}\to\mathrm{B}}^{\mathrm{penalty}} \lesssim 1.0$ (i. e. almost no penalty), the 'WN default' decoding result would be class B in the given example, as it represents the longer segment. However, in case of stricter penalties $\left(P_{\mathrm{A}\to\mathrm{B}}^{\mathrm{penalty}} < 0.869\right)$, the highest total probability is provided when the entire trial is modeled with HMM-A and hence, class A would be predicted.

A.2. Supplementary tables

Table A.1: Full results for comparison of Bakis model vs. unconstrained model.

Subject	Session	LFTD				
		Bakis	unconstr.	superior	t	p
S1	1	80.7 (1.9)	77.4 (2.3)	Bakis (***)	9.145	3.83E-13
	2	78.8 (3.4)	76.0 (3.6)	Bakis (***)	3.889	1.31E-04
S2	1	44.0 (5.5)	42.4 (5.6)	none	1.311	0.097
	2	45.7 (5.7)	45.7 (5.4)	none	0.000	0.500
	3	40.2 (3.7)	40.0 (3.4)	none	0.193	0.424
	4	28.8 (4.1)	28.6 (3.7)	none	0.177	0.430
S3	1	64.6 (3.7)	61.7 (4.4)	Bakis (***)	3.715	2.29E-04
	2	80.1 (2.7)	71.3 (3.3)	Bakis (***)	14.917	0.000
S4	1	71.4 (2.8)	70.5 (2.8)	none	1.256	0.107
	2	69.7 (4.1)	68.4 (4.1)	none	1.599	0.058
Average		60.4 (19.0)	58.2 (17.4)	Bakis (*)†	2.665†	0.013†

Subject	Session	HG				
		Bakis	unconstr.	superior	t	p
S1	1	97.9 (0.6)	97.1 (0.7)	Bakis (***)	4.719	7.69E-06
	2	92.4 (2.0)	91.4 (2.3)	Bakis (*)	1.827	0.036
S2	1	35.9 (3.8)	34.2 (3.3)	Bakis (*)	1.857	0.034
	2	45.6 (3.9)	48.1 (4.8)	unconstr. (**)	-2.346	0.011
	3	36.3 (4.3)	36.5 (4.3)	none	-0.181	0.429
	4	36.4 (4.7)	37.1 (5.6)	none	-0.529	0.300
S3	1	85.1 (2.2)	84.1 (2.7)	none	1.573	0.061
	2	90.5 (1.8)	86.5 (2.7)	Bakis (***)	6.711	4.47E-09
S4	1	83.5 (2.7)	81.5 (3.2)	Bakis (**)	2.600	0.006
	2	83.1 (1.9)	78.9 (2.0)	Bakis (***)	8.535	3.91E-12
Average		68.7 (26.4)	67.5 (25.3)	none†	1.758†	0.056†

†Significance analysis for differences in the averages is carried out with one-sample one-sided t-tests on the performance differences $\mathrm{Acc}^{\mathrm{Bakis}} - \mathrm{Acc}^{\mathrm{unconstr.}}$ (alternative hypothesis: $\mathrm{Acc}^{\mathrm{Bakis}} - \mathrm{Acc}^{\mathrm{unconstr.}} > 0$) in the individual datasets. The critical t-value for these tests (significance level $\alpha = 0.05$) is $t_c = 1.812$.

Table A.3: Continuous decoding accuracies P for all datasets without exclusion of pause and artifact segments. Different from the results shown in Table A.2, parameter optimization has been performed on training data that also includes rest episodes here.

Subject	Session	p	Δp	P [%] (std.)	ΔP [%]	ΔMatches	ΔSubst.	ΔIns.	ΔDel.
S1	B1B2	-44	-3	61.61 (1.78)	-1.77	-4.9	0.0	-1.0	7.9
	B2B1	-62	-6	68.80 (1.20)	-0.25	-6.7	0.1	-5.6	6.3
S2	B1B2	-40	-5	34.43 (1.79)	-1.31	-13.5	-2.7	-8.8	15.0
	B2B1	-34	-4	28.33 (0.89)	+1.90	-7.8	-1.2	-13.4	8.3
S3	B1B2	-28	-6	38.92 (0.94)	-0.60	-22.2	-12.1	-19.4	32.5
	B2B1	-34	-4	59.73 (0.96)	+4.16	-3.8	-1.8	-11.1	-7.3

Table A.4: Difference in count of matches M and decoding errors (substitutions S, insertions I and deletions D) between Session PK (Bi-Gram component only) and No PK.

	Sub-block 1				Sub-block 2				Sub-block 3			
		Errors				Errors				Errors		
Dataset	M	S	I	D	M	S	I	D	M	S	I	D
S1 - B1B2	0.3	-3.1	-6.5	2.8	1.4	-4.0	-3.2	2.6				
S1 - B2B1	-2.3	-3.0	-11.8	5.3	0.0	0.0	-0.4	0.0	0.9	-0.8	-0.7	-0.1
S2 - B1B2	4.9	-3.2	-0.9	-1.3	5.3	-5.2	-8.1	-0.1	0.4	-2.5	-2.1	2.1
S2 - B2B1	9.4	-6.9	-0.3	-2.6	3.9	-1.2	0.3	-2.7	14.4	5.0	10.8	-19.3
S3 - B1B2	16.5	-15.3	3.9	-2.0	9.0	-11.8	-8.8	2.8	6.9	-3.9	1.3	-2.1
S3 - B2B1	16.5	-15.4	-2.8	-1.1	2.8	-4.6	-2.2	1.3	1.1	-0.9	0.1	-0.2
Ø	7.6	-7.8	-3.1	0.2	3.7	-4.5	-3.7	0.6	4.7	-0.6	1.9	-3.9

Table A.2: Continuous decoding accuracies P for all datasets without exclusion of pause and artifact segments (training still performed on data without pause segments). Performance values are averaged across 10 runs with differing seeds for model initialization (standard deviation in brackets). Difference in decoding accuracy ΔP and count of insertions ΔIns. refer to the results with removed pause/artifacts segments (compare Table 4.14).

Subject	Session	P [%] (std.)	ΔP [%]	ΔIns.
S1	B1B2	63.37 (1.10)	-3.59	+ 7.1
	B2B1	69.05 (1.24)	-4.36	+15.7
S2	B1B2	35.75 (1.73)	-4.91	+16.0
	B2B1	26.42 (1.30)	-9.76	+29.7
S3	B1B2	39.52 (1.29)	-4.17	+14.6
	B2B1	55.57 (0.84)	-4.68	+16.9

Table A.5: Decoding accuracies Acc (in %) for picture category decoding using an HMM or SVM classifier. Reported t-values correspond to a two-sample one-sided t-test (H_0 : $\text{Acc}^{\text{HMM}} = \text{Acc}^{\text{SVM}}$). The rejection interval for the null hypothesis is $-\infty < t \leq -3.319$ for $\alpha = 0.001$ (alternative hypothesis: $\text{Acc}^{\text{SVM}} > \text{Acc}^{\text{HMM}}$).

	Decoding accuracy Acc		
Subject	HMM	SVM	t-value
ECoG 1	78.20 (1.90)	84.86 (1.51)	-12.27
ECoG 2	80.33 (1.15)	83.55 (1.65)	- 7.16
MEG 1	57.19 (2.45)	71.77 (2.82)	-17.45
MEG 2	63.41 (3.18)	75.33 (2.04)	-14.11

A.3. HTK

Data format

Features need to be converted into HTK-compatible file format. The definition of HTK files [126] allows for the use of user defined features by specifying a specific flag in the header. The full header is 12 bytes long and is composed of:

`nSamples` number of samples in file (4-byte integer),

`sampPeriod` sample period in 100 ns units (4-byte integer),

`sampSize` number of bytes per sample (2-byte integer),

`parmKind` a code indicating the sample kind (2-byte integer).

Here, `parmKind` encodes the type of features in a 6-bit code and uses the remaining bits to specify additional properties. For the purpose of converting data into HTK-compatible format, only the 6-bit code is relevant. In particular, one uses the option '9' that refers to "USER - user defined sample kind" [126]. Consequently, the binary representation of the required parameter is: 0000 0000 0000 1001 (10bit modifier, 6bit type).

To allow for convenient use within HTK subroutines, individual files are created for each of the trials in the dataset. Filenames follow a strict pattern to encode the class labels required for training the models:

$$\texttt{trial.}\underbrace{\texttt{xxxx}}_{\text{trial number}}.\underbrace{\texttt{x}}_{\text{label}}.\underbrace{\texttt{asd}}_{\text{arb. file ending}}$$

After the header, all data values are stored sequentially as 32bit (4 byte) float values. For means of compatibility with multiple channels, one simply passes:

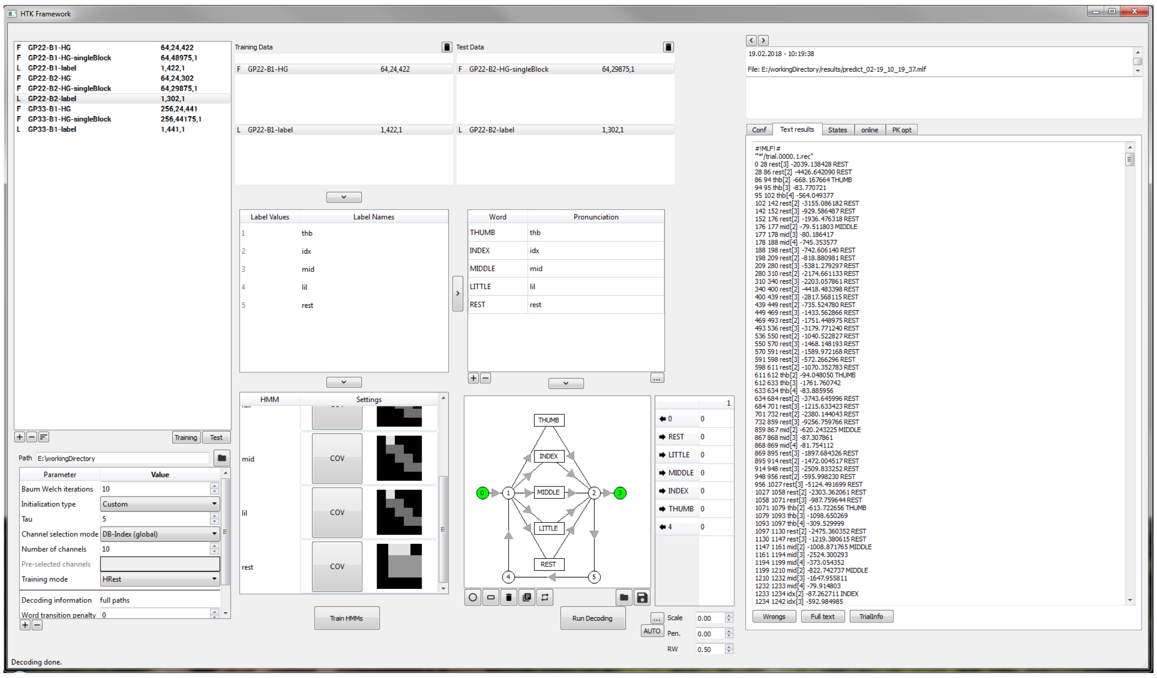
$$\texttt{sampSize} = 4 \cdot C$$

to the header, whereby C denotes the number of channels to be used.

A C++ implementation of data conversion and file storage in HTK-format is given in Code Example A.1.

Code Example A.1 C++ implementation of a data conversion routine for file storage in HTK-compatible format. Features of all trials are stored in `data`. Corresponding labels must be given in `label`. Note that the dimensions of the dataset (i. e. number of channels, samples, and trials) are private members of the DataIO class.

```cpp
bool DataIO::writeTrialsToHTK(float *data, float sPeriod, const QList<int> &label,
                              const QDir &saveDirectory) const
{
    QFile file;
    QDataStream out;
    out.setFloatingPointPrecision(QDataStream::SinglePrecision);


    // binary form 0000 0000 0000 1001 (10bit modifier, 6bit type)
    quint16 type = 1;
    type <<= 3;
    type |= quint16(1);


    int smplPerTrial  = _nbChannels*_nbSamples;


    for(int tr=0; tr<_nbTrials; ++tr)
    {
        // compose file name
        QString fileName("trial.");
        for(int i=0; i< (tr ? 3-(int)log10(double(tr)) : 3); ++i)
            fileName.append("0");
        fileName.append(QString::number(tr) + "." + QString::number(label.at(tr)));
        file.setFileName(saveDirectory.absolutePath() + "/" + fileName + ".asd");
        if(!file.open(QIODevice::WriteOnly))
            return false;


        // write data
        out.setDevice(&file);

        // header
        out << qint32(_nbSamples) << qint32(sPeriod) << qint16(4*_nbChannels) << type;
        // data
        for(int el = 0; el<smplPerTrial; ++el)
            out << data[el+smplPerTrial*tr];


        file.close();
    }


    return true;
}
```



F	GP22-B1-HG	64,24,422
F	GP22-B1-HG-singleBlock	64,48975,1
L	GP22-B1-label	1,422,1
F	GP22-B2-HG	64,24,302
F	GP22-B2-HG-singleBlock	64,29875,1
L	GP22-B2-label	1,302,1
F	GP33-B1-HG	256,24,441
F	GP33-B1-HG-singleBlock	256,44175,1
L	GP33-B1-label	1,441,1

Training Data

F	GP22-B1-HG	64,24,422
L	GP22-B1-label	1,422,1

Test Data

F	GP22-B2-HG-singleBlock	64,29875,1
L	GP22-B2-label	1,302,1

Label Values	Label Names
1	thb
2	idx
3	mid
4	lil
5	rest

Word	Pronunciation
THUMB	thb
INDEX	idx
MIDDLE	mid
LITTLE	lil
REST	rest

Path E:\workingDirectory

Parameter	Value
Baum Welch iterations	10
Initialization type	Custom
Tau	5
Channel selection mode	DB-Index (global)
Number of channels	10
Pre-selected channels	
Training mode	HRest
Decoding information	full paths
Word transition penalty	0

HMM	Settings
mid	COV
lil	COV
rest	COV

	1
0	0
REST	0
LITTLE	0
MIDDLE	0
INDEX	0
THUMB	0
4	0

Scale	0.00
Pen.	0.00
RW	0.50

19.02.2018 - 10:19:38

File: E:/workingDirectory/results/predict_02-19_10_19_37.mlf

```
#!MLF!#
"*/trial.0000.1.rec"
0 28 rest[3] -2039.138428 REST
28 86 rest[2] -4426.642090 REST
86 94 thb[2] -668.167664 THUMB
94 95 thb[3] -83.770721
95 102 thb[4] -564.049377
102 142 rest[2] -3155.086182 REST
142 152 rest[3] -929.586487 REST
152 176 rest[2] -1936.476318 REST
176 177 mid[2] -79.511803 MIDDLE
177 178 mid[3] -80.186417
178 188 mid[4] -745.353577
188 198 rest[3] -742.606140 REST
198 209 rest[2] -818.880981 REST
209 280 rest[3] -5381.279297 REST
280 310 rest[2] -2174.661133 REST
310 340 rest[3] -2203.057861 REST
340 400 rest[2] -4418.483398 REST
400 439 rest[3] -2817.568115 REST
439 449 rest[2] -735.524780 REST
449 469 rest[3] -1433.562866 REST
469 493 rest[2] -1751.448975 REST
493 536 rest[3] -3179.771240 REST
536 550 rest[2] -1040.522827 REST
550 570 rest[3] -1468.148193 REST
570 591 rest[2] -1589.972168 REST
591 598 rest[3] -572.266296 REST
598 611 rest[2] -1070.352783 REST
611 612 thb[2] -94.048050 THUMB
612 633 thb[3] -1761.760742
633 634 thb[4] -83.885956
634 684 rest[2] -3743.645996 REST
684 701 rest[3] -1215.633423 REST
701 732 rest[2] -2380.144043 REST
732 859 rest[3] -9256.759766 REST
859 867 mid[2] -620.243225 MIDDLE
867 868 mid[3] -87.307861
868 869 mid[4] -81.754112
869 895 rest[3] -1897.684326 REST
895 914 rest[2] -1472.004517 REST
914 948 rest[3] -2509.833252 REST
948 956 rest[2] -595.998230 REST
956 1027 rest[3] -5124.491699 REST
1027 1058 rest[2] -2303.362061 REST
1058 1071 rest[3] -987.759644 REST
1071 1079 thb[2] -613.722656 THUMB
1079 1093 thb[3] -1098.650269
1093 1097 thb[4] -309.529999
1097 1130 rest[2] -2475.360352 REST
1130 1147 rest[3] -1219.380615 REST
1147 1161 mid[2] -1008.871765 MIDDLE
1161 1194 mid[3] -2524.300293
1194 1199 mid[4] -373.054352
1199 1210 mid[2] -822.742737 MIDDLE
1210 1232 mid[3] -1647.955811
1232 1233 mid[4] -79.914803
1233 1234 idx[2] -87.262711 INDEX
1234 1242 idx[3] -592.984985
```

Decoding done.

Figure A.2: User interface of the developed HTK framework. The underlying structure is illustrated in Figure 3.2. The screenshot shows an example of a typical setting for continuous decoding of finger movements, as used for example in Section 4.5.

Training

This is an example for a routine call of the HTK submodule HRest for model parameter training:

```
"HRest -S E:/workingDirectory/train_thb.scp -M E:/workingDirectory/trained
-i 10 E:/workingDirectory/init/thb"
```

This call of HRest performs ten iterations of Baum-Welch re-estimation for a specific HMM (here: the thumb model “thb”) as defined in a model template file. Examples of the required files are shown in the following for the “thb”-model.

- List of training data (’train_thb.scp’)

```
feats/trial.0001.1.asd
feats/trial.0006.1.asd
feats/trial.0011.1.asd
feats/trial.0017.1.asd
feats/trial.0024.1.asd
...
```

- (initialized) Model template ('thb')

```
~o <VecSize> 10 <USER>
~h "thb"
<BeginHMM>
<NumStates>5
<State> 2
<Mean> 10
3136.21 2446.58 2505.05 1824.3 5413.02 2187.67 1983.41 4260.62 1916.68 1713.21
<Variance> 10
505809 143546 176180 71954.6 1.18188e+06 78470.7 96493.9 588626 381804 251940
<State> 3
<Mean> 10
4095.01 2634.9 2673.09 1765.76 6725.16 2107.72 1951.24 4385.82 2004.38 1736.09
<Variance> 10
361721 141101 141436 74521.7 1.65427e+06 76059.1 77123.1 474614 635137 319198
<State> 4
<Mean> 10
4328.4 2456.41 2525.94 1641.07 6638.58 1871.19 1700.58 3688.89 2148.22 1837.89
<Variance> 10
313106 129275 156810 71333.3 1.36088e+06 111306 62262.1 422508 307468 349991
<TransP> 5
0 1 0 0 0
0 0.5 0.5 0 0
0 0 0.5 0.5 0
0 0 0 0.5 0.5
0 0 0 0 0
<EndHMM>
```

Continuous decoding

The HTK routine HVite is used for the purpose of continuous recognition. An exemplary program call is shown in the following:

```
"HVite -w E:/workingDirectory/zeroGram -S E:/workingDirectory/test.scp -l '*'
-H E:/workingDirectory/trained/trainedHMMs
-i E:/workingDirectory/results/prediction.mlf
-f -p 0 E:/workingDirectory/pseudoDic E:/workingDirectory/HMMmonoPhoneList"
```

In addition to the trained models (output of HRest) and a list containing all test data files (analogue to training list), the following files are generated:

- Dictionary ('pseudoDic')

```
THUMB thb
INDEX idx
MIDDLE mid
LITTLE lil
REST rest
```

- Word network / language model ('zeroGram')

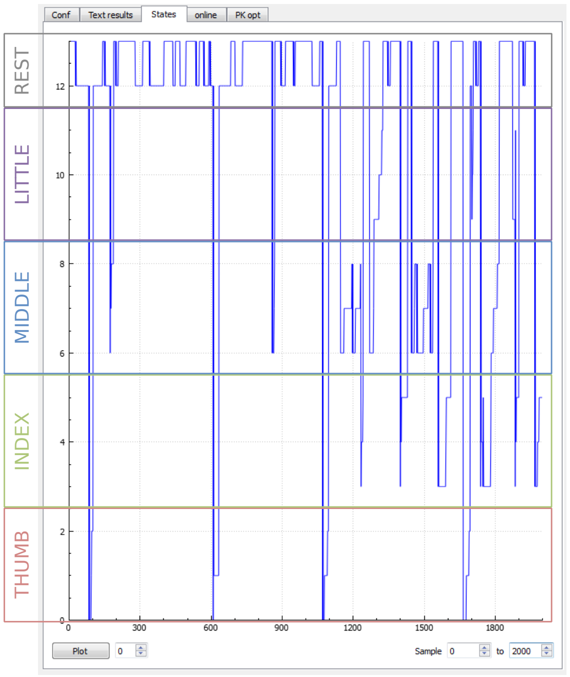
```
VERSION=1.0
N=11 L=15
I=0 W=!NULL
I=1 W=!NULL
I=2 W=!NULL
I=3 W=!NULL
I=4 W=!NULL
I=5 W=!NULL
I=6 W=REST
I=7 W=LITTLE
I=8 W=MIDDLE
I=9 W=INDEX
I=10 W=THUMB
J=0 S=4 E=1 l=0
J=1 S=5 E=4 l=0
J=2 S=2 E=5 l=0
J=3 S=10 E=2 l=0
J=4 S=1 E=10 l=0
J=5 S=9 E=2 l=0
J=6 S=1 E=9 l=0
J=7 S=8 E=2 l=0
J=8 S=1 E=8 l=0
J=9 S=7 E=2 l=0
J=10 S=1 E=7 l=0
J=11 S=6 E=2 l=0
J=12 S=1 E=6 l=0
J=13 S=2 E=3 l=0
J=14 S=0 E=1 l=0
```

PK incorporation

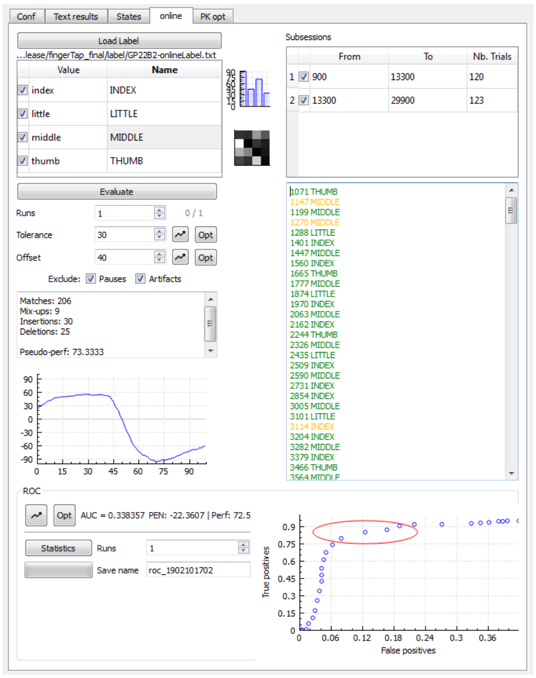
Prior knowledge can be incorporated conveniently using the dictionary and word network functionalities of HTK. Exemplary files are presented below. Especially word network specification is quite tedious. To facilitate easier construction of complex networks and to illustrate probability modifications, an editor widget has been implemented (Figure A.4) that allows for simple *what you see is what you mean* editing of word networks.

- Dictionary

```
index idx rest
little lil rest
middle mid rest
thumb thb rest
```



(a) State sequence



...lease/fingerTap_final/label/GP22B2-onlineLabel.txt

Value	Name
✓ index	INDEX
✓ little	LITTLE
✓ middle	MIDDLE
✓ thumb	THUMB

Runs 1 0 / 1
Tolerance 30
Offset 40
Exclude: ✓ Pauses ✓ Artifacts

```
Matches: 206
Mix-ups: 9
Insertions: 30
Deletions: 25

Pseudo-perf: 73.3333
```

Subsessions

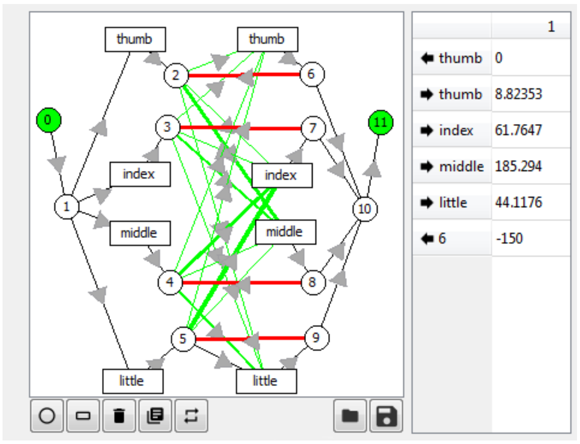
		From	To	Nb. Trials
1	✓	900	13300	120
2	✓	13300	29900	123

```
1071 THUMB
1147 MIDDLE
1199 MIDDLE
1270 MIDDLE
1288 LITTLE
1401 INDEX
1447 MIDDLE
1560 INDEX
1665 THUMB
1777 MIDDLE
1874 LITTLE
1970 INDEX
2063 MIDDLE
2162 INDEX
2244 THUMB
2326 MIDDLE
2435 LITTLE
2509 INDEX
2590 MIDDLE
2731 INDEX
2854 INDEX
3005 MIDDLE
3101 LITTLE
3114 INDEX
3204 INDEX
3282 MIDDLE
3379 INDEX
3466 THUMB
3564 MIDDLE
```

AUC = 0.338357 PEN: -22.3607 | Perf: 72.5
Runs 1
Save name roc_1902101702

(b) Evaluation

Figure A.3: Different options for result visualization in the HTK framework. (a) Plot of the (likeliest) state sequence for the decoded data segment. States of all individual HMMs (i. e. thumb, index, ..., rest) are concatenated for visualization (model correspondence indicated on the left). (b) Evaluation of predictions.



	1
⬅ thumb	0
➡ thumb	8.82353
➡ index	61.7647
➡ middle	185.294
➡ little	44.1176
⬅ 6	-150

Figure A.4: Word network editor showing a network with bi-gram structure. The table on the right lists all connection from/to the node ’2’ and their corresponding probability weightings. Negative values indicate penalized connections (red lines), whereas transitions with positive values get rewarded (green lines).

- Word network / language model (word network as shown in Figure A.4)

```
VERSION=1.0
N=20 L=38
I=0 W=!NULL
I=1 W=!NULL
I=2 W=!NULL
I=3 W=!NULL
I=4 W=!NULL
I=5 W=!NULL
I=6 W=!NULL
I=7 W=!NULL
I=8 W=!NULL
I=9 W=!NULL
I=10 W=!NULL
I=11 W=!NULL
I=12 W=little
I=13 W=middle
I=14 W=index
I=15 W=thumb
I=16 W=little
I=17 W=middle
I=18 W=index
I=19 W=thumb
J=0 S=10 E=11 l=0
J=1 S=6 E=10 l=0
J=2 S=6 E=2 l=-150
J=3 S=19 E=6 l=0
J=4 S=7 E=10 l=0
J=5 S=7 E=3 l=-150
J=6 S=18 E=7 l=0
J=7 S=8 E=10 l=0
J=8 S=8 E=4 l=-150
J=9 S=17 E=8 l=0
J=10 S=9 E=10 l=0
J=11 S=9 E=5 l=-150
J=12 S=16 E=9 l=0
J=13 S=5 E=19 l=26.6667
J=14 S=5 E=18 l=226.667
J=15 S=5 E=17 l=46.6667
J=16 S=5 E=16 l=0
J=17 S=4 E=16 l=118.31
J=18 S=4 E=17 l=4.22535
J=19 S=4 E=18 l=156.338
J=20 S=4 E=19 l=21.1268
...
J=37 S=0 E=1 l=0
```

Own publications – on the topic of BCI

Journal articles

T. Pfeiffer, R. T. Knight, G. Rose: Hidden Markov model-based continuous decoding of finger movements with prior knowledge incorporation using bi-gram models. **Biomedical Physics & Engineering Express**, 4(2):025007, 2018.

N. Heinze*, T. Pfeiffer*, M. A. Schoenfeld, G. Rose: Schätzung von Erkennungsraten auf Elektrokortikografie Daten mit Hilfe von vollständig nicht-invasiven MEG Messungen. **Klinische Neurophysiologie**, 48(01):40–43, 2017. (*equal contribution)

T. Pfeiffer, N. Heinze, R. Frysch, L. Y. Deouell, M. A. Schoenfeld, R. T. Knight, G. Rose: Extracting duration information in a picture category decoding task using hidden Markov models. **Journal of Neural Engineering**, 13(2):026010, 2016.

T. Wissel, T. Pfeiffer, R. Frysch, R. T. Knight, E. F. Chang, H. Hinrichs, J. W. Rieger, G. Rose: Hidden Markov model and support vector machine based decoding of finger movements using electrocorticography. **Journal of Neural Engineering**, 10(5):056020, 2013.

Conference contributions

M. Will, T. Pfeiffer, N. Heinze, A. Schoenfeld, G. Rose: SSVEP controlled BCI inferring complex tasks from low-level-commands. 62. Wissenschaftliche Jahrestagung der DGKN, 2018. *accepted.*
C. Reichert, N. Heinze, T. Pfeiffer, H. Hinrichs: BMI Accuracy Improvement by Detecting Errors from EEG and MEG Recordings. 3rd Image-Guided Interventions Conference (IGIC) & Fokus Neuroradiologie, 2017.
T. Pfeiffer, R. T. Knight, G. Rose: Word networks for BCI decoding purposes. Proceedings of the Sixth International Brain-Computer Interface Meeting: BCI Past, Present, and Future, 160, 2016.
N. Heinze*, T. Pfeiffer*, A. Schoenfeld, G. Rose: P140. Towards an estimation of ECoG decoding results based on fully non-invasive MEG acquisition, Clinical Neurophysiology, 126(8):e156-e157, 2015. (*equal contribution)
T. Pfeiffer, N. Heinze, A. Schoenfeld, G. Rose: Investigating information content from different brain areas for single trial MEG decoding. 7th International IEEE/EMBS Conference on Neural Engineering, pp. 1–44, 2015.
T. Pfeiffer, N. Heinze, E. Gerber, L. Y. Deouell, J. Parvizi, R. T. Knight, G. Rose: Decoding of picture category and presentation duration - preliminary results of a combined ECoG and MEG study. Proceedings of the 6th International Brain-Computer Interface Conference 2014, pp. ID 042–1, 2014.
T. Pfeiffer, N. Heinze, G. Rose: Influence of MEG data from different brain areas on decoding picture category information. 1st Conference on Image-Guided Interventions (IGIC), pp. 63–63, 2014.

Own publications – other topics

Journal articles

S. Bannasch, R. Frysch, T. Pfeiffer, G. Warnecke, G. Rose: Time separation technique: Accurate solution for 4D C-Arm-CT perfusion imaging using a temporal decomposition model. **Medical Physics**, 45(3):1080–1092, 2018.

Conference contributions

S. Waldherr, R. Frysch, T. Pfeiffer, T. Jakuszeit, S. Zeng, G. Rose: A numerical evaluation of state reconstruction methods for heterogeneous cell populations. 2015 European Control Conference. pp. 2931–2936, 2015.
S. Bannasch, T. Pfeiffer, G. Warnecke, G. Rose: Acceleration of a regularized Algebraic Reconstruction Technique evaluated with a simulation of computed tomography. IMA Conference on Numerical Methods for Simulation 2015, Mathematical Institute, University of Oxford ; abstracts book and delegate list. p. 17, 2015.
R. Frysch, T. Pfeiffer, S. Bannasch, S. Serowy, S. Gugel, M. Skalej, G. Rose: C-arm perfusion imaging with a fast penalized maximum-likelihood approach. Proceedings of SPIE Medical imaging conference, p. 90332M, 2014.
S. Bannasch, G. Warnecke, R. Frysch, T. Pfeiffer, G. Rose: An implicit optimization approach for the Kaczmarz method applied to algebraic reconstruction techniques for computed tomography. 4th IMA Conference on Numerical Linear Algebra and Optimisation. p. 10, 2014.
R. Frysch, T. Pfeiffer, G. Rose: Scalable OpenCL accelerated multi-GPU projection for cone beam computed tomography with a highly accurate separable footprint method. Interventional neuroradiology : journal of peritherapeutic neuroradiology, surgical procedures and related neurosciences. 19, 204–246, 2013.
T. Pfeiffer, R. Frysch, S. Gugel, G. Rose: ML reconstruction of cone-beam projections acquired by a flat-panel rotational X-ray device. Proceedings of SPIE Medical imaging conference, p. 86682V, 2013.