

---

# Deep learning with convolutional neural networks for brain mapping and decoding of movement-related information from the human EEG

---

Short title: *Convolutional neural networks in EEG analysis*

Keywords: *Electroencephalography, EEG analysis, machine learning, end-to-end learning, brain-machine interface (BCI), brain-computer interface (BMI), model interpretability, brain mapping*

Robin Tibor Schirrmeister<sup>a,b</sup>, Jost Tobias Springenberg<sup>c,b</sup>, Lukas Dominique Josef Fiederer<sup>a,b,d</sup>, Martin Glasstetter<sup>a,b</sup>, Katharina Eggensperger<sup>e,b</sup>, Michael Tangermann<sup>f,b</sup>, Frank Hutter<sup>e,b</sup>, Wolfram Burgard<sup>g,b</sup>, and Tonio Ball<sup>1a,b</sup>

<sup>a</sup>*Intracranial EEG and Brain Imaging lab, Epilepsy Center – University of Freiburg*

<sup>b</sup>*BrainLinks-BrainTools Cluster of Excellence, University of Freiburg*

<sup>c</sup>*Machine Learning Lab, Computer Science Dept., University of Freiburg*

<sup>d</sup>*Neurobiology and Biophysics, Faculty of Biology, University of Freiburg*

<sup>e</sup>*Machine Learning for Automated Algorithm Design Lab, Computer Science Dept., University of Freiburg*

<sup>f</sup>*Brain State Decoding Lab, Computer Science Dept., University of Freiburg*

<sup>g</sup>*Autonomous Intelligent Systems Lab, Computer Science Dept., University of Freiburg*

June 11, 2018

<sup>1</sup>Corresponding author: tonio.ball@uniklinik-freiburg.de

## Abstract

Deep learning with convolutional neural networks (deep ConvNets) has revolutionized computer vision through end-to-end learning, i.e. learning from the raw data. Now, there is increasing interest in using deep ConvNets for end-to-end EEG analysis. However, little is known about many important aspects of how to design and train ConvNets for end-to-end EEG decoding, and there is still a lack of techniques to visualize the informative EEG features the ConvNets learn.

Here, we studied deep ConvNets with a range of different architectures, designed for decoding imagined or executed movements from raw EEG. Our results show that recent advances from the machine learning field, including batch normalization and exponential linear units, together with a cropped training strategy, boosted the deep ConvNets decoding performance, reaching or surpassing that of the widely-used filter bank common spatial patterns (FBCSP) decoding algorithm. While FBCSP is designed to use spectral power modulations, the features used by ConvNets are not fixed *a priori*. Our novel methods for visualizing the learned features demonstrated that ConvNets indeed learned to use spectral power modulations in the alpha, beta and high gamma frequencies. These methods also proved useful as a technique for spatially mapping the learned features, revealing the topography of the causal contributions of features in different frequency bands to decoding the movement classes.

Our study thus shows how to design and train ConvNets to decode movement-related information from the raw EEG without handcrafted features and highlights the potential of deep ConvNets combined with advanced visualization techniques for EEG-based brain mapping.

## 1 Introduction

Machine-learning techniques allow to extract information from electroencephalographic (EEG) recordings of brain activity and therefore play a crucial role in several important EEG-based research and application areas. For example, machine-learning techniques are a central component of many EEG-based brain-computer interface (BCI) systems for clinical applications. Such systems already allowed, for example, persons with severe paralysis to communicate (Nijboer et al., 2008), to draw pictures (Münßinger et al., 2010), and to control telepresence robots (Tonin et al., 2011). Such systems may also facilitate stroke rehabilitation (Ramos-Murguialday et al., 2013) and may be used in the treatment of epilepsy (Gadhouni et al., 2016) (for more examples of potential clinical applications, see Moghimi et al. (2013)). Furthermore, machine-learning techniques for the analysis of brain signals, including the EEG, are increasingly recognized as novel tools for neuroscientific inquiry (Das et al., 2010; Knops et al., 2009; Kurth-Nelson et al., 2016; Stansbury et al., 2013).

However, despite many examples of impressive progress, there is still room for considerable improvement with respect to the accuracy of information extraction from the EEG and hence, an interest in transferring advances from the area of machine learning to the field of EEG decoding and BCI. A recent, prominent example of such an advance in machine learning is the application of convolutional neural networks (ConvNets), particularly in computer vision tasks. Thus, first studies have started to investigate the potential of ConvNets for brain-signal decoding ((Antoniades et al., 2016; Bashivan et al., 2016; Cecotti and Graser, 2011; Hajinorozi et al., 2016; Lawhern et al., 2016; Liang et al., 2016; Manor et al., 2016; Manor and Geva, 2015; Page et al., 2016; Ren and Wu, 2014; Sakhavi et al., 2015; Shamwell et al., 2016; Stober, 2016; Stober et al., 2014; Sun et al., 2016; Tabar and Halici, 2017; Tang et al., 2017; Thodoroff et al., 2016; Wang et al., 2013), see Supplementary Section A.1 for more details on these studies). Still, several important methodological questions on EEG analysis with ConvNets remain, as detailed below and addressed in the present study.

ConvNets are artificial neural networks that can learn local patterns in data by using convolutions as their key component (also see Section 2.3). ConvNets vary in the number of convolutional layers, ranging from shallow architectures with just one convolutional layer such as in a successful speech recognition ConvNet (Abdel-Hamid et al., 2014) over deep ConvNets with multiple consecutive convolutional layers (Krizhevsky et al., 2012) to very deep architectures with more than 1000 layers as in the case of the recently developed residual networks (He et al., 2015). Deep ConvNets can first extract local, low-level features from the raw input and then increasingly more global and high level features in deeper layers. For example, deep ConvNets can learn to detect increasingly complex visual features (e.g., edges, simple shapes, complete objects) from raw images. Over the past years, deep ConvNets have become highly successful in many application areas, such as in computer vision and speech recognition, often outperforming previous state-of-the-art methods (we refer to LeCun et al. (2015) and Schmidhuber (2015) for recent reviews). For example, deep ConvNets reduced the error rates on the ImageNet image-recognition challenge, where 1.2 million images must be classified into 1000 different classes, from above 26% to below 4% within 4 years (He et al., 2015; Krizhevsky et al., 2012). ConvNets also reduced error rates in recognizing speech, e.g., from English news broadcasts (Sainath et al., 2015a; Sercu et al., 2016); however, in this field, hybrid models combining ConvNets with other machine-learning components, notably recurrent networks, and deep neural networks without convolutions are also competitive (Li and Wu, 2015; Sainath et al., 2015b; Sak et al., 2015). Deep ConvNets also contributed to the spectacular success of AlphaGo, an artificial intelligence that beat the world champion in the game of Go (Silver et al., 2016).

An attractive property of ConvNets that was leveraged in many previous applications is that

they are well suited for end-to-end learning, i.e., learning from the raw data without any *a priori* feature selection. End-to-end learning might be especially attractive in brain-signal decoding, as not all relevant features can be assumed to be known *a priori*. Hence, in the present study we have investigated how ConvNets of different architectures and designs can be used for end-to-end learning of EEG recorded in human subjects.

The EEG signal has characteristics that make it different from inputs that ConvNets have been most successful on, namely images. In contrast to two-dimensional static images, the EEG signal is a dynamic time series from electrode measurements obtained on the three-dimensional scalp surface. Also, the EEG signal has a comparatively low signal-to-noise ratio, i.e., sources that have no task-relevant information often affect the EEG signal more strongly than the task-relevant sources. These properties could make learning features in an end-to-end fashion fundamentally more difficult for EEG signals than for images. Thus, the existing ConvNets architectures from the field of computer vision need to be adapted for EEG input and the resulting decoding accuracies rigorously evaluated against more traditional feature extraction methods. For that purpose, a well-defined baseline is crucial, i.e., a comparison against an implementation of a standard EEG decoding method validated on published results for that method. In light of this, in the present study we addressed two key questions:

- What is the impact of ConvNet *design choices* (e.g., the overall network architecture or other design choices such as the type of non-linearity used) on the decoding accuracies?
- What is the impact of ConvNet *training strategies* (e.g., training on entire trials or crops within trials) on decoding accuracies?

To address these questions, we created three ConvNets with different architectures, with the number of convolutional layers ranging from 2 layers in a “shallow” ConvNet over a 5-layer deep ConvNet up to a 31-layer residual network (ResNet). Additionally, we also created a hybrid ConvNet from the deep and shallow ConvNets. As described in detail in the methods section, these architectures were inspired both from existing “non-ConvNet” EEG decoding methods, which we embedded in a ConvNet, as well as from previously published successful ConvNet solutions in the image processing domain (for example, the ResNet architecture recently won several image recognition competitions ([He et al., 2015](#)). All architectures were adapted to the specific requirements imposed by the analysis of multi-channel EEG data. To address whether these ConvNets can reach competitive decoding accuracies, we performed a statistical comparison of their decoding accuracies to those achieved with decoding based on filter bank common spatial patterns (FBCSP, [Ang et al. \(2008\)](#); [Chin et al. \(2009\)](#)), a method that is widely used in EEG decoding and has won several EEG decoding competitions such as BCI Competition IV 2a and 2b. We analyzed the offline decoding performance on two suitable motor decoding EEG datasets (see Section 2.7 for details). In all cases, we used only minimal preprocessing to conduct a fair end-to-end comparison of ConvNets and FBCSP.

In addition to the role of the overall network architecture, we systematically evaluated a range of important design choices. We focussed on alternatives resulting from recent advances in machine-learning research on deep ConvNets. Thus, we evaluated potential performance improvements by using dropout as a novel regularization strategy ([Srivastava et al., 2014](#)), intermediate normalization by batch normalization ([Ioffe and Szegedy, 2015](#)) or exponential linear units as a recently proposed activation function ([Clevert et al., 2016](#)). A comparable analysis of the role of deep ConvNet design choices in EEG decoding is currently lacking.

In addition to the global architecture and specific design choices which together define the “structure” of ConvNets, another important topic that we address is how a given ConvNet should be trained

on the data. As with architecture and design, there are several different methodological options and choices with respect to the training process, such as the optimization algorithm (e.g., Adam (Kingma and Ba, 2014), Adagrad (Duchi et al., 2011), etc.), or the sampling of the training data. Here, we focused on the latter question of sampling the training data as there is usually, compared to current computer vision tasks with millions of samples, relatively little data available for EEG decoding. Therefore, we evaluated two sampling strategies, both for the deep and shallow ConvNets: training on whole trials or on multiple crops of the trial, i.e., on windows shifted through the trials. Using multiple crops holds promise as it increases the amount of training examples, which has been crucial to the success of deep ConvNets. Using multiple crops has become standard procedure for ConvNets for image recognition (see (He et al., 2015; Howard, 2013; Szegedy et al., 2015)), but the usefulness of cropped training has not yet been examined in EEG decoding.

In addition to the problem of achieving good decoding accuracies, a growing corpus of research tackles the problem of understanding what ConvNets learn (see Yeager (2016) for a recent overview). This direction of research may be especially relevant for neuroscientists interested in using ConvNets — insofar as they want to understand what features in the brain signal discriminate the investigated classes. Here we present two novel methods for *feature visualization* that we used to gain insights into our ConvNet learned from the neuronal data.

We concentrated on EEG band power features as a target for visualizations. Based on a large body of literature on movement-related spectral power modulations (Chatrian et al., 1959; Pfurtscheller and Aranibar, 1977, 1978; Pfurtscheller and Berghold, 1989; Pfurtscheller et al., 1994; Toro et al., 1994), we had clear expectations which band power features should be discriminative for the different classes; thus our rationale was that visualizations of these band power features would be particularly useful to verify that the ConvNets are using actual brain signals. Also, since FBCSP uses these features too, they allowed us to directly compare visualizations for both approaches. Our first method can be used to show how much information about a specific feature is retained in the ConvNet in different layers, however it does not evaluate whether the feature causally affects the ConvNet outputs. Therefore, we designed our second method to directly investigate causal effects of the feature values on the ConvNet outputs. With both visualization methods, it is possible to derive topographic scalp maps that either show how much information about the band power in different frequency bands is retained in the outputs of the trained ConvNet or how much they causally affect the outputs of the trained ConvNet.

Addressing the questions raised above, in summary the main contributions of this study are as follows:

- We show for the first time that end-to-end deep ConvNets can reach accuracies at least in the same range as FBCSP for decoding movement-related information from EEG.
- We evaluate a large number of ConvNet design choices on an EEG decoding task, and we show that recently developed methods from the field of deep learning such as batch normalization and exponential linear units are crucial for reaching high decoding accuracies.
- We show that cropped training can increase the decoding accuracy of deep ConvNets and describe a computationally efficient training strategy to train ConvNets on a larger number of input crops per EEG trial.
- We develop and apply novel visualizations that highly suggest that the deep ConvNets learn to use the band power in frequency bands relevant for motor decoding (alpha, beta, gamma) with meaningful spatial distributions.

Thus, in summary, the methods and findings described in this study pave the way for a widespread application of deep ConvNets for EEG decoding both in clinical applications and neuroscientific research.

## 2 Methods

We first provide basic definitions with respect to brain-signal decoding as a supervised classification problem used in the remaining Methods section. This is followed by the principles of both filter bank common spatial patterns (FBCSP), the established baseline decoding method referred to throughout the present study, and of convolutional neural networks (ConvNets). Next, we describe the ConvNets developed for this study in detail, including the *design choices* we evaluated. Afterwards, the training of the ConvNets, including two *training strategies*, are described. Then we present two novel *visualizations of trained ConvNets* in Section 2.6. Datasets and preprocessing descriptions follow in Section 2.7. Details about statistical evaluation, software and hardware can be found in Supplementary Sections A.8 and A.9.

### 2.1 Definitions and notation

This section more formally defines how brain-signal decoding can be viewed as a supervised classification problem and includes the notation used to describe the methods.

#### 2.1.1 Input and labels

We assume that we are given one EEG data set per subject  $i$ . Each dataset is separated into labeled trials (time-segments of the original recording that each belong to one of several classes). Concretely, we are given datasets  $D^i = \{(X^1, y^1), \dots, (X^{N_i}, y^{N_i})\}$  where  $N_i$  denotes the total number of recorded trials for subject  $i$ . The input matrix  $X^j \in \mathbb{R}^{E \cdot T}$  of trial  $j$ ,  $1 \leq j \leq N_i$  contains the preprocessed signals of  $E$  recorded electrodes and  $T$  discretized time steps recorded per trial.

The corresponding class label of trial  $j$  is denoted by  $y^j$ . It takes values from a set of  $K$  class labels  $L$  that, in our case, correspond to the imagined or executed movements performed in each trial, e.g.:  $\forall y^j : y^j \in L = \{l_1 = \text{"Hand (Left)"}, l_2 = \text{"Hand (Right)"}, l_3 = \text{"Feet"}, l_4 = \text{"Rest"}\}$ .

#### 2.1.2 Decoder

The decoder  $f$  is trained on these existing trials such that it is able to assign the correct label to new unseen trials. Concretely, we aim to train the decoder to assign the label  $y^j$  to trial  $X^j$  using the output of a parametric classifier  $f(X^j; \theta) : \mathbb{R}^{E \cdot T} \rightarrow L$  with parameters  $\theta$ .

For the rest of this manuscript we assume that the classifier  $f(X^j; \theta)$  is represented by a standard machine-learning pipeline decomposed into two parts: A first part that extracts a (vector-valued) feature representation  $\phi(X^j; \theta_\phi)$  with parameters  $\theta_\phi$  — which could either be set manually (for hand designed features), or learned from the data; and a second part consisting of a classifier  $g$  with parameters  $\theta_g$  that is trained using these features, i.e.,  $f(X^j; \theta) = g(\phi(X^j; \theta_\phi), \theta_g)$ .

As described in detail in the following sections, it is important to note that FBCSP and ConvNets differ in how they implement this framework: in short, FBCSP has separated feature extraction and classifier stages, while ConvNets learn both stages jointly.

### 2.2 Filter bank common spatial patterns (FBCSP)

FBCSP (Ang et al., 2008; Chin et al., 2009) is a widely-used method to decode oscillatory EEG data, for example, with respect to movement-related information, i.e., the decoding problem we focus on in this study. FBCSP was the best-performing method for the BCI competition IV dataset 2a, which

we use in the present study (in the following called *BCI Competition Dataset*, see Section 2.7 for a short dataset description). FBCSP also won other similar EEG decoding competitions (Tangermann et al., 2012). Therefore, we consider FBCSP an adequate benchmark algorithm for the evaluation of the performance of ConvNets in the present study.

In the following, we explain the computational steps of FBCSP. We will refer to these steps when explaining our shallow ConvNet architecture (see Section 2.4.3), as it is inspired by these steps.

In a supervised manner, FBCSP computes spatial filters (linear combinations of EEG channels) that enhance class-discriminative band power features contained in the EEG. FBCSP extracts and uses these features  $\phi(X^j; \theta_\phi)$  (which correspond to the feature representation part in Section 2.1.2) to decode the label of a trial in several steps (we will refer back to these steps when we explain the shallow ConvNet):

1. **Bandpass filtering:** Different bandpass filters are applied to separate the raw EEG signal into different frequency bands.
2. **Epoching:** The continuous EEG signal is cut into trials as explained in Section 2.1.1.
3. **CSP computation:** Per frequency band, the common spatial patterns (CSP) algorithm is applied to extract spatial filters. CSP aims to extract spatial filters that make the trials discriminable by the power of the spatially filtered trial signal (see Koles et al. (1990); Ramoser et al. (2000); Blankertz et al. (2008) for more details on the computation). The spatial filters correspond to the learned parameters  $\theta_\phi$  in FBCSP.
4. **Spatial filtering:** The spatial filters computed in Step 2 are applied to the EEG signal.
5. **Feature construction:** Feature vectors  $\phi(X^j; \theta_\phi)$  are constructed from the filtered signals: Specifically, feature vectors are the log-variance of the spatially filtered trial signal for each frequency band and for each spatial filter.
6. **Classification:** A classifier is trained to predict per-trial labels based on the feature vectors.

For details on our FBCSP implementation, see Supplementary Section A.2.

## 2.3 Convolutional neural networks

In the following sections, we first explain the basic ideas of ConvNets. We then describe architectural choices for ConvNets on EEG, including how to represent the EEG input for a ConvNet, the three different ConvNet architectures used in this study and several specific design choices that we evaluated for these architectures. Finally, we describe how to train the ConvNets, including the description of a trial-wise and a cropped training strategy for our EEG data.

### 2.3.1 Basics

Generally, ConvNets combine two ideas useful for many learning tasks on natural signals, such as images and audio signals. These signals often have an inherent hierarchical structure (e.g., images typically consist of edges that together form simple shapes which again form larger, more complex shapes and so on). ConvNets can learn local non-linear features (through convolutions and nonlinearities) and represent higher-level features as compositions of lower level features (through multiple

layers of processing). In addition, many ConvNets use pooling layers which create a coarser intermediate feature representation and can make the ConvNet more translation-invariant. For further details see [LeCun et al. \(2015\)](#); [Goodfellow et al. \(2016\)](#); [Schmidhuber \(2015\)](#).

## 2.4 ConvNet architectures and design choices

### 2.4.1 Input representation

The first important decision for applying ConvNets to EEG decoding is how to represent the input  $X^j \in \mathbb{R}^{E \cdot T}$ . One possibility would be to represent the EEG as a time series of topographically organized images, i.e., of the voltage distributions across the (flattened) scalp surface (this has been done for ConvNets that get power spectra as input ([Bashivan et al., 2016](#)). However, EEG signals are assumed to approximate a linear superposition of spatially global voltage patterns caused by multiple dipolar current sources in the brain ([Nunez and Srinivasan, 2006](#)). Unmixing of these global patterns using a number of spatial filters is therefore typically applied to the whole set of relevant electrodes as a basic step in many successful examples of EEG decoding ([Ang et al., 2008](#); [Blankertz et al., 2008](#); [Rivet et al., 2009](#)).

In this view, all relevant EEG modulations are global in nature, due to the physical origin of the non-invasive EEG and hence there would be no obvious hierarchical compositionality of local and global EEG modulations *in space*. In contrast, there is an abundance of evidence that the EEG is organized across multiple time scales, such as in nested oscillations ([Canolty et al., 2006](#); [Monto et al., 2008](#); [Schack et al., 2002](#); [Vanhatalo et al., 2004](#)) involving both local and global modulations *in time*. In light of this, we designed ConvNets in a way that they can learn spatially global unmixing filters in the entrance layers, as well as temporal hierarchies of local and global modulations in the deeper architectures. To this end we represent the input as a 2D-array with the number of time steps as the width and the number of electrodes as the height. This approach also significantly reduced the input dimensionality compared with the “EEG-as-an-image” approach.

### 2.4.2 Deep ConvNet for raw EEG signals

To tackle the task of EEG decoding we designed a deep ConvNet architecture inspired by successful architectures in computer vision, as for example described in [Krizhevsky et al. \(2012\)](#). The requirements for this architecture are as follows: We want a model that is able to extract a wide range of features and is not restricted to specific feature types ([Hertel et al., 2015](#)). We were interested in such a generic architecture for two reasons: 1) we aimed to uncover whether such a generic ConvNet designed only with minor expert knowledge can reach competitive accuracies, and, 2) to lend support to the idea that standard ConvNets can be used as a general-purpose tool for brain-signal decoding tasks. As an aside, keeping the architecture generic also increases the chances that ConvNets for brain decoding can directly profit from future methodological advances in deep learning.

Our deep ConvNet had four convolution-max-pooling blocks, with a special first block designed to handle EEG input (see below), followed by three standard convolution-max-pooling blocks and a dense softmax classification layer (see Figure 1). The first convolutional block was split into two convolutional layers in order to better handle the large number of input channels — one input channel per electrode compared to three input channels (one per color) in rgb-images. The convolution was split into a first convolution across time and a second convolution across space (electrodes); each filter in these steps has weights for all electrodes (like a CSP spatial filter) and for the filters of the preceding temporal convolution (like any standard intermediate convolutional layer). Since there is

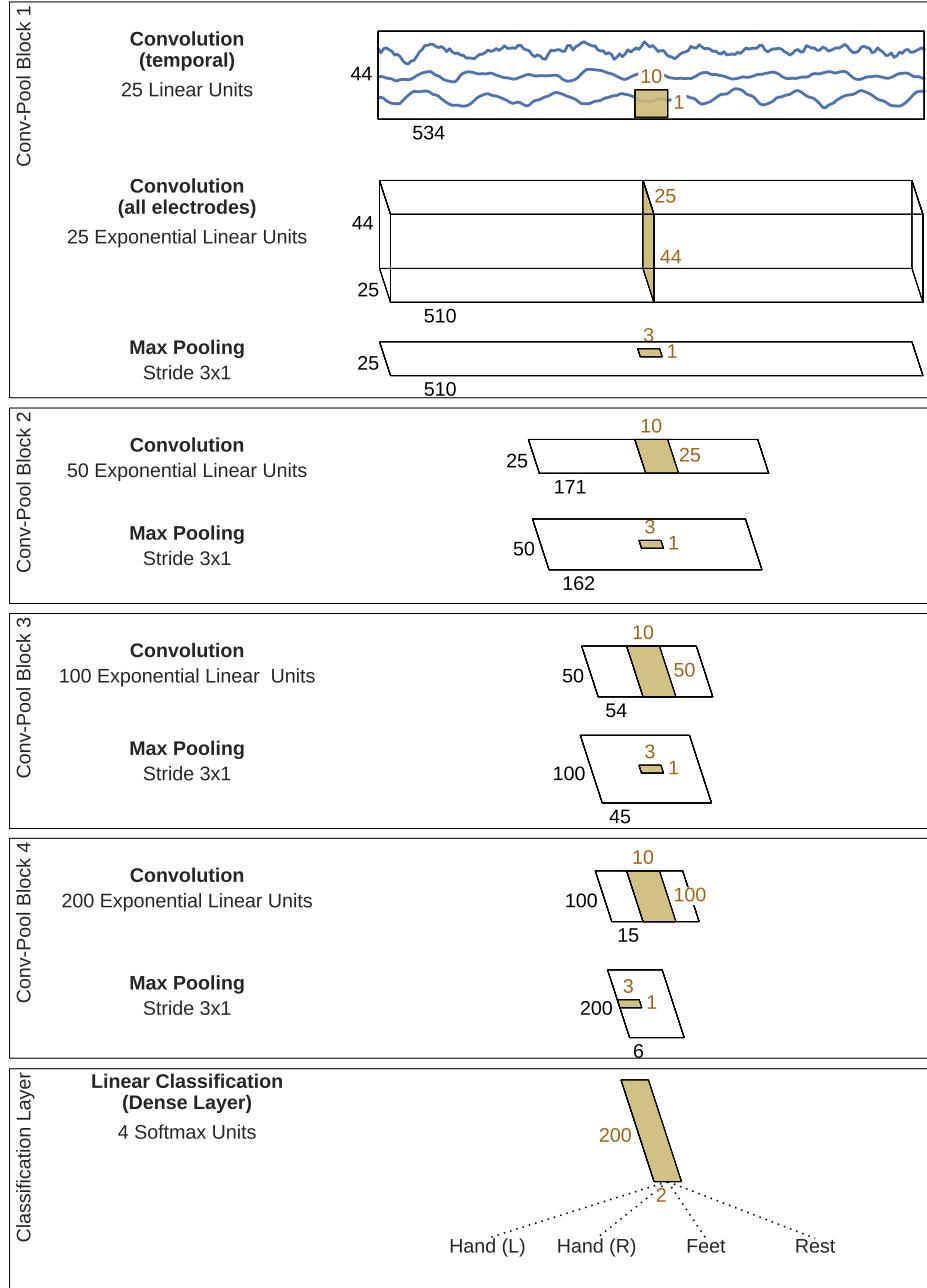


Figure 1: **Deep ConvNet architecture.** EEG input (at the top) is progressively transformed towards the bottom, until the final classifier output. Black cuboids: inputs/feature maps; brown cuboids: convolution/pooling kernels. The corresponding sizes are indicated in black and brown, respectively. Note that in this schematics, proportions of maps and kernels are only approximate.

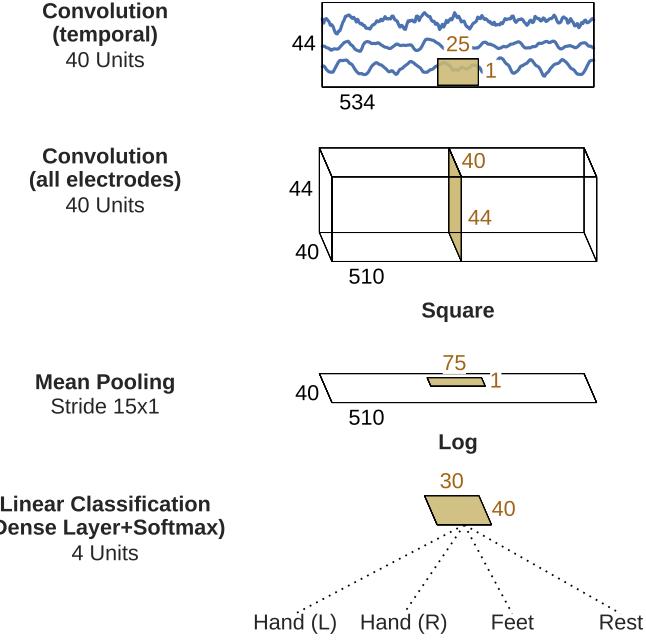


Figure 2: **Shallow ConvNet architecture.** Conventions as in Figure 1.

no activation function in between the two convolutions, they could in principle be combined into one layer. Using two layers however implicitly regularizes the overall convolution by forcing a separation of the linear transformation into a combination of two (temporal and spatial) convolutions. This splitted convolution was evaluated against a single-step convolution in our experiments (see Section 2.4.4).

We used exponential linear units (ELUs,  $f(x) = x$  for  $x > 0$  and  $f(x) = e^x - 1$  for  $x \leq 0$  (Clevert et al., 2016)) as activation functions (we also evaluated Rectified Linear Units (ReLUs,  $f(x) = \max(x, 0)$ ), as a less recently proposed alternative, see Section 2.4.4).

### 2.4.3 Shallow ConvNet for raw EEG signals

We also designed a more shallow architecture referred to as shallow ConvNet, inspired by the FBCSP pipeline (see Figure 2), specifically tailored to decode band power features. The transformations performed by the shallow ConvNet are similar to the transformations of FBCSP (see Section 2.2). Concretely, the first two layers of the shallow ConvNet perform a temporal and a spatial convolution, as in the deep ConvNet. These steps are analogous to the bandpass and CSP spatial filter steps in FBCSP. In contrast to the deep ConvNet, the temporal convolution of the shallow ConvNet had a larger kernel size (25 vs 10), allowing a larger range of transformations in this layer (smaller kernel sizes for the shallow ConvNet led to lower accuracies in preliminary experiments). After the two convolutions of the shallow ConvNet, a squaring nonlinearity, a mean pooling layer and a logarithmic activation function followed; together these steps are analogous to the trial log-variance computation in FBCSP (we note that these steps were not used in the deep ConvNet). In contrast to FBCSP, the

shallow ConvNet embeds all the computational steps in a single network, and thus all steps can be optimized jointly (see Section 2.5). Also, due to having several pooling regions within one trial, the shallow ConvNet can learn a temporal structure of the band power changes within the trial, which was shown to help classification in prior work (Sakhavi et al., 2015).

#### 2.4.4 Design choices for deep and shallow ConvNet

For both architectures described above we evaluated several design choices. We evaluated architectural choices which we expect to have a potentially large impact on the decoding accuracies and/or from which we hoped to gain insights into the behavior of the ConvNets. Thus, for the deep ConvNet, we compared the design aspects listed in Table 1.

Design aspect	Our choice	Variants	Motivation
Activation functions	ELU	square, ReLU	We expected these choices to be sensitive to the type of feature (e.g., signal phase or power), since squaring and mean pooling results in mean power (given a zero-mean signal). Different features may play different roles in the low-frequency components vs. the higher frequencies (see Section 2.7).
Pooling mode	max	mean	
Regularization and intermediate normalization	Dropout + batch normalization + a new tied loss function (explanations see text)	Only batch normalization, only dropout, neither of both, no tied loss	We wanted to investigate whether recent deep learning advances improve accuracies and check how much regularization is required.
Factorized temporal convolutions	One 10x1 convolution per convolutional layer	Two 6x1 convolutions per convolutional layer	Factorized convolutions are used by other successful ConvNets (see Szegedy et al. (2015))
Splitted vs one-step convolution	Splitted convolution in first layer (see Section 2.4.2)	one-step convolution in first layer	Factorizing convolution into spatial and temporal parts may improve accuracies for the large number of EEG input channels (compared with three rgb color channels of regular image datasets).

Table 1: **Evaluated design choices.** Design choices we evaluated for our convolutional networks. “Our choice” are the choices we used when evaluating ConvNets in the remainder of this manuscript, e.g., vs FBCSP. Note that these design choices have not been evaluated in prior work, see Supplementary Section A.1

In the following, we give additional details for some of these aspects. Batch normalization standardizes intermediate outputs of the network to zero mean and unit variance for a batch of training examples (Ioffe and Szegedy, 2015). This is meant to facilitate the optimization by keeping the inputs of layers closer to a normal distribution during training. We applied batch normalization, as recommended in the original paper (Ioffe and Szegedy, 2015), to the output of convolutional layers before the nonlinearity. Dropout randomly sets some inputs for a layer to zero in each training update. It is meant to prevent co-adaption of different units and can be seen as analogous to training an ensemble of networks. We drop out the inputs to all convolutional layers after the first with a probability of 0.5. Finally, our new tied loss function is designed to further regularize our cropped training (see Section 2.5.4 for an explanation).

We evaluated the same design aspects for the shallow ConvNet, except for the following differences:

- The baseline methods for the activation function and pooling mode were chosen as “squaring nonlinearity” and “mean pooling”, motivation is given in Section 2.4.3.
- We did not include factorized temporal convolutions into the comparison, as the longer kernel lengths of the shallow ConvNet make these convolutions less similar to other successful ConvNets anyways.
- We additionally compared the logarithmic nonlinearity after the pooling layer with a square root nonlinearity to check if the logarithmic activation inspired by FBCSP is better than traditional L2-pooling.

#### 2.4.5 Hybrid ConvNet

Besides the individual design choices for the deep and shallow ConvNet, a natural question to ask is whether both ConvNets can be combined into a single ConvNet. Such a hybrid ConvNet could profit from the more specific feature extraction of the shallow ConvNet as well as from the more generic feature extraction of the deep ConvNet. Therefore, we also created a hybrid ConvNet by fusing both networks after the final layer. Concretely, we replaced the four-filter softmax classification layers of both ConvNets by 60- and 40-filter ELU layers for the deep and shallow ConvNet respectively. The resulting 100 feature maps were concatenated and used as the input to a new softmax classification layer. We retrained the whole hybrid ConvNet from scratch and did not use any pretrained deep or shallow ConvNet parameters.

#### 2.4.6 Residual ConvNet for raw EEG signals

In addition to the shallow and deep ConvNets, we evaluated another network architecture: Residual networks (ResNets), a ConvNet architecture that recently won several benchmarks in the computer vision field (He et al., 2015). ResNets typically have a very large number of layers and we wanted to investigate whether similar networks with more layers also result in good performance in EEG decoding. ResNets add the input of a convolutional layer to the output of the same layer, to the effect that the convolutional layer only has to learn to output a residual that changes the previous layers output (hence the name residual network). This allows ResNets to be successfully trained with a much larger number of layers than traditional convolutional networks (He et al., 2015). Our residual blocks are the same as described in the original paper (see Figure 3).

Our ResNet used exponential linear unit activation functions (Clevert et al., 2016) throughout the network (same as the deep ConvNet) and also starts with a splitted temporal and spatial convolution (same as the deep and shallow ConvNets), followed by 14 residual blocks, mean pooling and a final softmax dense classification layer (for further details, see Supplementary Section A.3).

### 2.5 ConvNet training

In this section, we first give a definition of how ConvNets are trained in general. Second, we describe two ways of extracting training inputs and training labels from the EEG data, which result in a trialwise and a cropped training strategy.

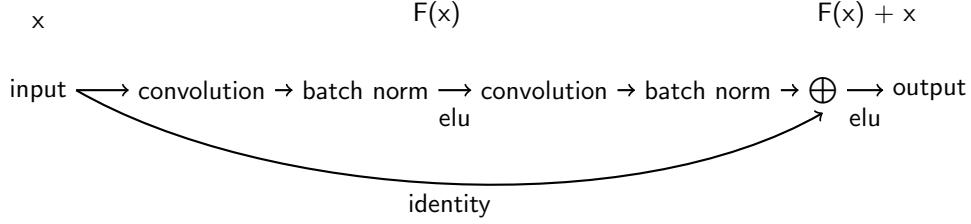


Figure 3: **Residual Block.** Residual block used in the ResNet architecture and as described in original paper (He et al., 2015), see Figure 2) with identity shortcut option A, except using ELU instead of ReLU nonlinearities. See Section 2.4.6 for explanation.

### 2.5.1 Definition

To train a ConvNet, all parameters (all weights and biases) of the ConvNet are trained jointly. Formally, in our supervised classification setting, the ConvNet computes a function from input data to one real number per class,  $f(X^j; \theta) : \mathbb{R}^{E \cdot T} \rightarrow \mathbb{R}^K$ , where  $\theta$  are the parameters of the function,  $E$  the number of electrodes,  $T$  the number of timesteps and  $K$  the number of possible output labels. To use ConvNets for classification, the output is typically transformed to conditional probabilities of a label  $l_k$  given the input  $X^j$  using the softmax function:  $p(l_k|f(X^j; \theta)) = \frac{\exp(f_k(X^j; \theta))}{\sum_{k=1}^K \exp(f_k(X^j; \theta))}$ . In our case, since we train per subject, the softmax output gives us a subject-specific conditional distribution over the  $K$  classes. Now we can train the entire ConvNet to assign high probabilities to the correct labels by minimizing the sum of the per-example losses:

$$\theta^* = \arg \min_{\theta} \sum_{j=1}^N \text{loss}\left(y^j, p(l_k|f_k(X^j; \theta))\right) \quad (1)$$

, where

$$\text{loss}\left(y^j, p(l_k|f_k(X^j; \theta))\right) = \sum_{k=1}^K -\log\left(p(l_k|f_k(X^j; \theta))\right) \cdot \delta(y^j = l_k) \quad (2)$$

is the negative log likelihood of the labels. As is common for training ConvNets, the parameters are optimized via mini-batch stochastic gradient descent using analytical gradients computed via backpropagation (see LeCun et al. (2015) for an explanation in the context of ConvNets and Section 2.5.5 in this manuscript for details on the optimizer used in this study).

This ConvNet training description is connected to our general EEG decoding definitions from Section 2.1 as follows. The function that the ConvNet computes can be viewed as consisting of a feature extraction function and a classifier function: The feature extraction function  $\phi(X^j; \theta_\phi)$  with parameters  $\theta_\phi$  is computed by all layers up to the penultimate layer. The classification function  $g(\phi(X^j; \theta_\phi), \theta_g)$  with parameters  $\theta_g$ , which uses the output of the feature extraction function as input, is computed by the final classification layer. In this view, one key advantage of ConvNets becomes clear: With the joint optimization of both functions, a ConvNet learns both, a descriptive feature representation for the task as well as a discriminative classifier. This is especially useful with large datasets, where it is more likely that the ConvNet learns to extract useful features and does

not just overfit to noise patterns. For EEG data, learning features can be especially valuable since there may be unknown discriminative features or at least discriminative features that are not used by more traditional feature extraction methods such as FBCSP.

### 2.5.2 Input and labels

In this study, we evaluated two ways of defining the input examples and target labels that the ConvNet is trained on. First, a trial-wise strategy that uses whole trials as input and per-trial labels as targets. Second, a cropped training strategy that uses crops, i.e., sliding time windows within the trial as input and per-crop labels as targets (where the label of a crop is identical to the label of the trial the crop was extracted from).

### 2.5.3 Trial-wise training

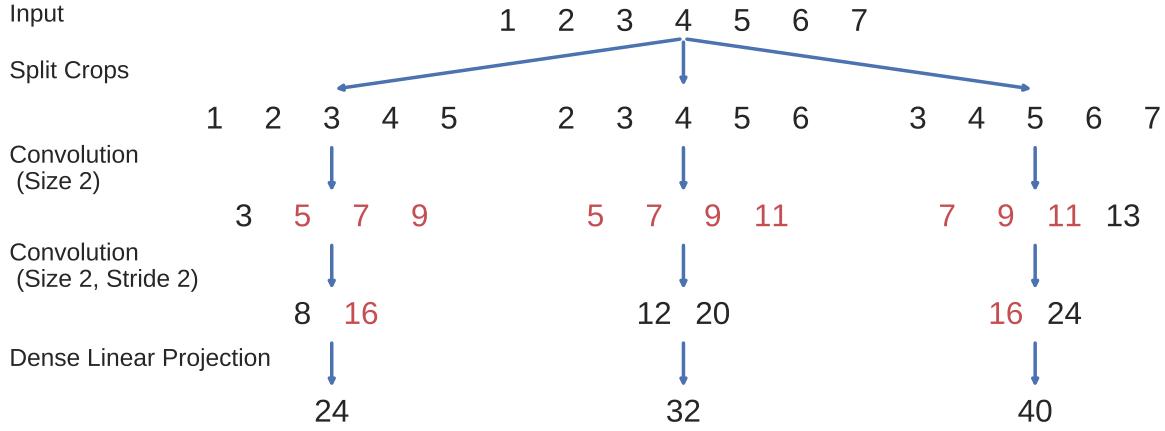
The standard trial-wise training strategy uses the whole duration of the trial and is therefore similar to how FBCSP is trained. For each trial, the trial signal is used as input and the corresponding trial label as target to train the ConvNet. In our study, for both datasets we had 4.5-second trials (from 500 ms before trial start cue until trial end cue, as that worked best in preliminary experiments) as the input to the network. This led to 288 training examples per subject for the BCI Competition Dataset and about 880 training examples per subject on the High-Gamma Dataset after their respective train-test split.

### 2.5.4 Cropped training

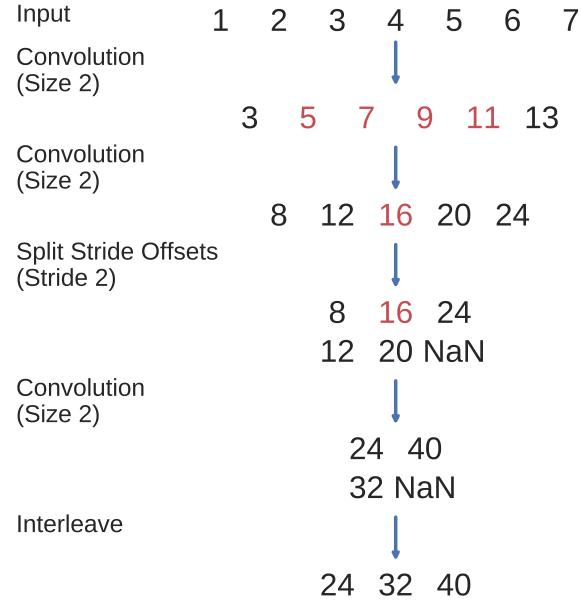
The cropped training strategy uses crops, i.e., sliding input windows within the trial, which leads to many more training examples for the network than the trial-wise training strategy. We adapted this strategy from convolutional neural networks for object recognition in images, where using multiple crops of the input image is a standard procedure to increase decoding accuracy (see for example [He et al. \(2015\)](#) and [Szegedy et al. \(2015\)](#)).

In our study, we used crops of about 2 seconds as the input. We adopt a cropping approach, which leads to the largest possible number of crops by creating one crop per sample (by sample, we mean a timestep in our EEG trial time series). More formally, given an original trial  $X^j \in \mathbb{R}^{E \cdot T}$  with  $E$  electrodes and  $T$  timesteps, we create a set of crops with crop size  $T'$  as timeslices of the trial:  $C^j = \{X_{1..E,t..t+T'}^j | t \in 1..T - T'\}$ . All of these  $T - T'$  crops are new training data examples for our decoder and will get the same label  $y^j$  as the original trial.

This aggressive cropping has the aim to force the ConvNet into using features that are present in all crops of the trial, since the ConvNet can no longer use the differences between crops and the global temporal structure of the features in the complete trial. We collected crops starting from 0.5 seconds before trial start (first crop from 0.5 seconds before to 1.5 seconds after trial start), with the last crop ending 4 seconds after the trial start (which coincides with the trial end, so the last crop starts 2 seconds before the trial and continues to the trial end). Overall, this resulted in 625 crops and therefore 625 label predictions per trial. The mean of these 625 predictions is used as the final prediction for the trial during the test phase. During training, we compute a loss for each prediction. Therefore, cropped training increases our training set size by a factor of 625, albeit with highly correlated training examples. Since our crops are smaller than the trials, the ConvNet input size is also smaller (from about 1000 input samples to about 500 input samples for the 250 Hz sampling rate), while all other hyperparameters stay the same.



(a) Naïve implementation by first splitting the trial into crops and passing the crops through the ConvNet independently.



(b) Optimized implementation, computing the outputs for each crop in a single forward pass. Strides in the original ConvNet are handled by separating intermediate results that correspond to different stride offsets, see the split stride offsets step. NaNs are only needed to pad all intermediate outputs to the same size and are removed in the end. The split stride step can simply be repeated in case of further layers with stride. We interleave the outputs only after the final predictions, also in the case of ConvNets with more layers.

**Figure 4: Multiple-crop prediction used for cropped training.** In this toy example, a trial with the sample values 1,2,3,4,5,6,7 is cut into three crops of length 5 and these crops are passed through a convolutional network with two convolutional layers and one dense layer. The convolutional layers both have kernel size 2, the second one additionally uses a stride of 2. Filters for both layers and the final dense layer have values 1,1. Red indicates intermediate outputs that were computed multiple times in the naïve implementation. Note that both implementations result in the same final outputs.

To reduce the computational load from the increased training set size, we decoded a group of neighboring crops together and reused intermediate convolution outputs. This idea has been used in the same way to speed up ConvNets that make predictions for each pixel in an image (Giusti et al., 2013; Nasse et al., 2009; Sermanet et al., 2014; Shelhamer et al., 2016). In a nutshell, this method works by providing the ConvNet with an input that contains several crops and computing the predictions for all crops in a single forward pass (see Figure 4 for an explanation). This cropped training method leads to a new hyperparameter: the number of crops that are processed at the same time. The larger this number of crops, the larger the speedup one can get (upper bounded by the size of one crop, see Giusti et al. (2013) for a more detailed speedup analysis on images), at the cost of increased memory consumption. A larger number of crops that are processed at the same time during training also implies parameter updates from gradients computed on a larger number of crops from the same trial during mini-batch stochastic gradient descent, with the risk of less stable training. However, we did not observe substantial accuracy decreases when enlarging the number of simultaneously processed crops (this stability was also observed for images (Shelhamer et al., 2016)) and in the final implementation we processed about 500 crops in one pass, which corresponds to passing the ConvNet an input of about 1000 samples, twice the 500 samples of one crop. Note that this method only results in exactly the same predictions as the naïve method when using valid convolutions (i.e., no padding). For padded convolutions (which we use in the residual network described in Section 2.4.6), the method no longer results in the same predictions, so it cannot be used to speed up predictions for individual samples anymore. However, it can still be used if one is only interested in the average prediction for a trial as we are in this study.

To further regularize ConvNets trained with cropped training, we designed a new objective function, which penalizes discrepancies between predictions of neighboring crops. In this *tied sample loss function*, we added the cross-entropy of two neighboring predictions to the usual loss of negative log likelihood of the labels. So, denoting the prediction  $p(l_k|f_k(X_{t..t+T'}^j; \theta))$  for crop  $X_{t..t+T'}^j$  from time step  $t$  to  $t+T'$  by  $p_{f,k}(X_{t..t+T'}^j)$ , the loss now also depends on the prediction for the next crop  $p_{f,k}(X_{t..t+T'+1}^j)$  and changes from equation 2 to:

$$\begin{aligned} \text{loss}(y^j, p_{f,k}(X_{t..t+T'}^j)) = & \sum_{k=1}^K -\log(p_{f,k}(X_{t..t+T'}^j)) \cdot \delta(y^j = l_k) + \\ & \sum_{k=1}^K -\log(p_{f,k}(X_{t..t+T'}^j)) \cdot p_{f,k}(X_{t..t+T'+1}^j) \end{aligned} \quad (3)$$

This is meant to make the ConvNet focus on features which are stable for several neighboring input crops.

### 2.5.5 Optimization and early stopping

As optimization method, we used Adam (Kingma and Ba, 2014) together with a specific early stopping method, since this consistently yielded good accuracies in our experiments. For details on Adam and our early stopping strategy, see Supplementary Section A.4.

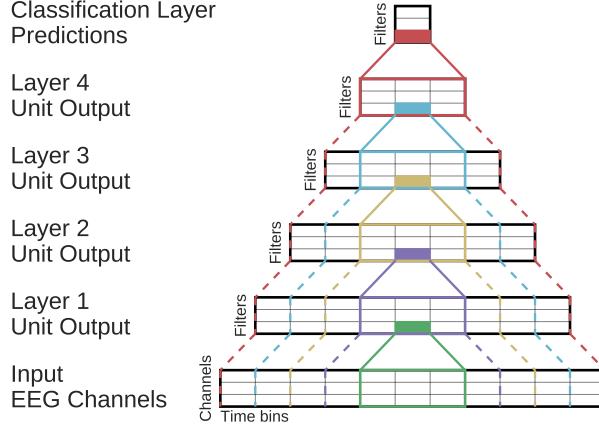


Figure 5: **ConvNet Receptive Fields Schema.** Showing the outputs, inputs and receptive fields of one unit per layer. Colors indicate different units. Filled rectangles are individual units, solid lines indicate their direct input from the layer before. Dashed lines indicate the corresponding receptive field in all previous layers including the original input layer. The receptive field of a unit contains all inputs that are used to compute the unit’s output. The receptive fields get larger with increasing depth of the layer. Note that this is only a schema and exact dimensions are not meaningful in this figure.

## 2.6 Visualization

### 2.6.1 Correlating Input Features and Unit Outputs: Network Correlation Maps

As described in the Introduction, currently there is a great interest in understanding how ConvNets learn to solve different tasks. To this end, methods to visualize functional aspects of ConvNets can be helpful and the development of such methods is an active area of research. Here, we wanted to delineate what brain-signal features the ConvNets used and in which layers they extracted these features. The most obvious restriction on possible features is that units in individual layers of the ConvNet can only extract features from samples that they have “seen”, i.e., from their so-called *receptive field* (see Figure 5). A way to further narrow down the possibly used features is to use domain-specific prior knowledge and to investigate whether known class-discriminative features are learned by the ConvNet. Then it is possible to compute a feature value for all receptive fields of all individual units for each of these class-discriminative features and to measure how much this feature affects the unit output, for example by computing the correlation between feature values and unit outputs.

In this spirit, we propose input-feature unit-output correlation maps as a method to visualize how networks learn spectral amplitude features. It is known that the amplitudes, for example of the alpha, beta and gamma bands, provide class-discriminative information for motor tasks (Ball et al., 2008; Pfurtscheller, 1981; Pfurtscheller and Aranibar, 1979). Therefore, we used the mean envelope values for several frequency bands as feature values. We correlated these values inside a receptive field of a unit, as a measure of its total spectral amplitude, with the corresponding unit outputs to gain insight into how much these amplitude features are used by the ConvNet. Positive

or negative correlations that systematically deviate from those found in an untrained net imply that the ConvNet learned to create representations that contain more information about these features than before training.

A limitation of this approach is that it does not distinguish between correlation and causation (i.e., whether the change in envelope caused the change in the unit output, or whether another feature, itself correlated to the unit output, caused the change). Therefore, we propose a second visualization method, where we perturbed the amplitude of existing inputs and observed the change in predictions of the ConvNets. This complements the first visualization and we refer to this method as input-perturbation network-prediction correlation map. By using artificial perturbations of the data, they provide insights in whether changes in specific feature amplitudes cause the network to change its outputs. For details on the computation of both NCM methods and a ConvNet-independent visualization, see Supplementary Section A.5.

## 2.7 Data sets and preprocessing

We evaluated decoding accuracies on two EEG datasets, a smaller public dataset (BCI Competition IV dataset 2a) for comparing to previously published accuracies and a larger new dataset acquired in our lab for evaluating the decoding methods with a larger number of training trials (approx. 880 trials per subject, compared to 288 trials in the public set). For details on the datasets, see Supplementary Section A.6.

### 2.7.1 EEG preprocessing and evaluating different frequency bands

We only minimally preprocessed the datasets to allow the ConvNets to learn any further transformations themselves. In addition to the full-bandwidth ( $0-f_{end}$ -Hz) dataset, we analyzed data high-pass filtered above 4 Hz (which we call  $4-f_{end}$ -Hz dataset). Filtering was done with a causal 3rd order Butterworth filter. We included the  $4-f_{end}$ -Hz dataset since the highpass filter should make it less probable that either the networks or FBCSP would use class-discriminative eye movement artifacts to decode the behavior classes, as eye movements generate most power in such low frequencies (Gratton, 1998). We included this analysis as for the BCI Competition Dataset, special care to avoid decoding eye-related signals was requested from the publishers of the dataset (Brunner et al., 2008). For details on other preprocessing steps, see Supplementary Section A.7.

Dataset	Frequency range [Hz]	FBCSP	Deep ConvNet	Shallow ConvNet	Hybrid ConvNet	Residual ConvNet
BCIC	0–38	68.0	+2.9	+5.7*	+3.6	-0.3
BCIC	4–38	67.8	+2.3	+4.1	-1.6	-7.0*
HGD	0–125	91.2	+1.3	-1.9	+0.6	-2.3*
HGD	4–125	90.9	+0.5	+3.0*	+1.5	-1.1
Combined	0– $f_{end}$	82.1	+1.9*	+1.1	+1.8	-1.1
Combined	4– $f_{end}$	81.9	+1.2	+3.4**	+0.3	-3.5*

Table 2: **Decoding accuracy of FBCSP baseline as well as of the deep and shallow ConvNets.** FBCSP decoding accuracies and difference of deep and shallow ConvNet accuracies to FBCSP results are given in percent. BCIC: BCI Competition Dataset. HGD: High-Gamma Dataset. Frequency range is in Hz. Stars indicate statistically significant differences (p-values from Wilcoxon signed-rank test, \*:  $p < 0.05$ , \*\*:  $p < 0.01$ , no p-values were below 0.001).

## 3 Results

### 3.1 Validation of FBCSP baseline

#### Result 1 *FBCSP baseline reached same results as previously reported in the literature*

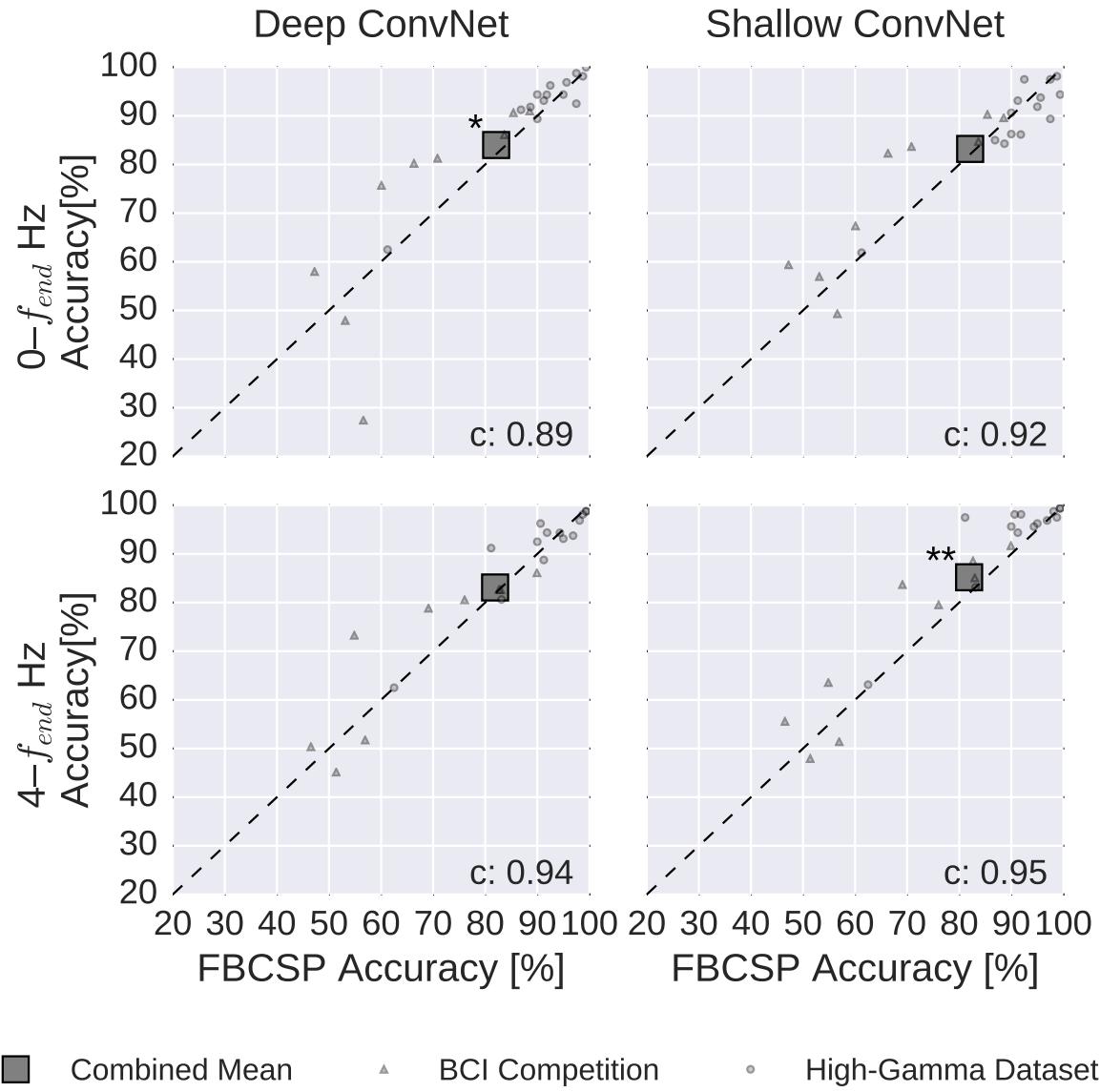
As a first step before moving to the evaluation of ConvNet decoding, we validated our FBCSP implementation, as this was the baseline we compared the ConvNets results against. To validate our FBCSP implementation, we compared its accuracies to those published in the literature for the BCI competition IV dataset 2a (called BCI Competition Dataset in the following) ([Sakhavi et al., 2015](#)). Using the same 0.5–2.5 s (relative to trial onset) time window, we reached an accuracy of 67.6%, statistically not significantly different from theirs (67.0%,  $p=0.73$ , Wilcoxon signed-rank test). Note however, that we used the full trial window for later experiments with convolutional networks, i.e., from 0.5–4 seconds. This yielded a slightly better accuracy of 67.8%, which was still not statistically significantly different from the original results on the 0.5–2.5 s window ( $p=0.73$ ). For all later comparisons, we use the 0.5–4 seconds time window on all datasets.

### 3.2 Architectures and design choices

#### Result 2 *ConvNets reached FBCSP accuracies*

Both the deep the shallow ConvNets, with appropriate design choices (see Result 5), reached similar accuracies as FBCSP-based decoding, with small but statistically significant advantages for the ConvNets in some settings. For the mean of all subjects of both datasets, accuracies of the shallow ConvNet on 0– $f_{end}$  Hz and for the deep ConvNet on 4– $f_{end}$  Hz were not statistically significantly different from FBCSP (see Figure 6 and Table 2). The deep ConvNet on 0– $f_{end}$  Hz and the shallow ConvNet on 4– $f_{end}$  Hz reached slightly higher (1.9% and 3.3% higher respectively) accuracies that were also statistically significantly different ( $p < 0.05$ , Wilcoxon signed-rank test). Note that all results in this section were obtained with cropped training, for a comparison of cropped and trial-wise training, see Section 3.3.

#### Result 3 *Confusion matrices for all decoding approaches were similar*



**Figure 6: FBCSP vs. ConvNet decoding accuracies.** Each small marker represents accuracy of one subject, the large square markers represent average accuracies across all subjects of both datasets. Markers above the dashed line indicate experiments where ConvNets performed better than FBCSP and opposite for markers below the dashed line. Stars indicate statistically significant differences between FBCSP and ConvNets (Wilcoxon signed-rank test,  $p < 0.05$ : \*,  $p < 0.01$ : \*\*,  $p < 0.001$  = \*\*\*). Bottom left of every plot: linear correlation coefficient between FBCSP and ConvNet decoding accuracies. Mean accuracies were very similar for ConvNets and FBCSP, the (small) statistically significant differences were in direction of the ConvNets.

	Hand (L) Hand (R)	Hand (L) Feet	Hand (L) Rest	Hand (R) Feet	Hand (R) Rest	Feet Rest
FBCSP	82	28	31	3	12	42
Deep	70	13	27	13	21	26
Shallow	99	3	34	5	37	73

Table 3: Decoding mistakes between class pairs. Results for the High-Gamma Dataset. Number of trials where one class was mistaken for the other for each decoding method, summed per class pair. The largest number of mistakes was between Hand(L) and Hand (R) for all three decoding methods, the second largest between Feet and Rest (on average across the three decoding methods). Together, these two class pairs accounted for more than 50% of all mistakes for all three decoding methods. In contrast, Hand (L and R) and Feet had a small number of mistakes irrespective of the decoding method used.

Confusion matrices for the High-Gamma Dataset on  $0-f_{end}$  Hz were very similar for FBCSP and both ConvNets (see Figure 7). The majority of all mistakes were due to discriminating between Hand (L) / Hand (R) and Feet / Rest, see Table 3. Seven entries of the confusion matrix had a statistically significant difference ( $p < 0.05$ , Wilcoxon signed-rank test) between the deep and the shallow ConvNet, in all of them the deep ConvNet performed better. Only two differences between the deep ConvNet and FBCSP were statistically significant ( $p < 0.05$ ), none for the shallow ConvNet and FBCSP. Confusion matrices for the BCI Competition Dataset showed a larger variability and hence a less consistent pattern, possibly because of the much smaller number of trials.

#### **Result 4** Hybrid ConvNets performed slightly, but statistically insignificantly, worse than deep ConvNets

The hybrid ConvNet performed similar, but slightly worse than the deep ConvNet, i.e., 83.8% vs 84.0% ( $p > 0.5$ , Wilcoxon signed-rank test) on the  $0-f_{end}$ -Hz dataset, 82.1% vs 83.1% ( $p > 0.9$ ) on the  $4-f_{end}$ -Hz dataset. In both cases, the hybrid ConvNet's accuracy was also not statistically significantly different from FBCSP (83.8% vs 82.1%,  $p > 0.4$  on  $0-f_{end}$  Hz, 82.1% vs 81.9%,  $p > 0.7$  on  $4-f_{end}$  Hz).

#### **Result 5** ConvNet design choices substantially affected decoding accuracies

In the following, results for all design choices are reported for all subjects from both datasets. For an overview of the different design choices investigated, and the motivation behind these choices, we refer to Section 2.4.4.

Batch normalization and dropout significantly increased accuracies. This became especially clear when omitting both simultaneously (see Figure 8a). Batch normalization provided a larger accuracy increase for the shallow ConvNet, whereas dropout provided a larger increase for the deep ConvNet. For both networks and for both frequency bands, the only statistically significant accuracy differences were accuracy decreases after removing dropout for the deep ConvNet on  $0-f_{end}$ -Hz data or removing batch normalization and dropout for both networks and frequency ranges ( $p < 0.05$ , Wilcoxon signed-rank test). Usage of tied loss did not affect the accuracies very much, never yielding statistically significant differences ( $p > 0.05$ ). Splitting the first layer into two convolutions had the strongest accuracy increase on the  $0-f_{end}$ -Hz data for the shallow ConvNet, where it is also the only statistically significant difference ( $p < 0.01$ ).

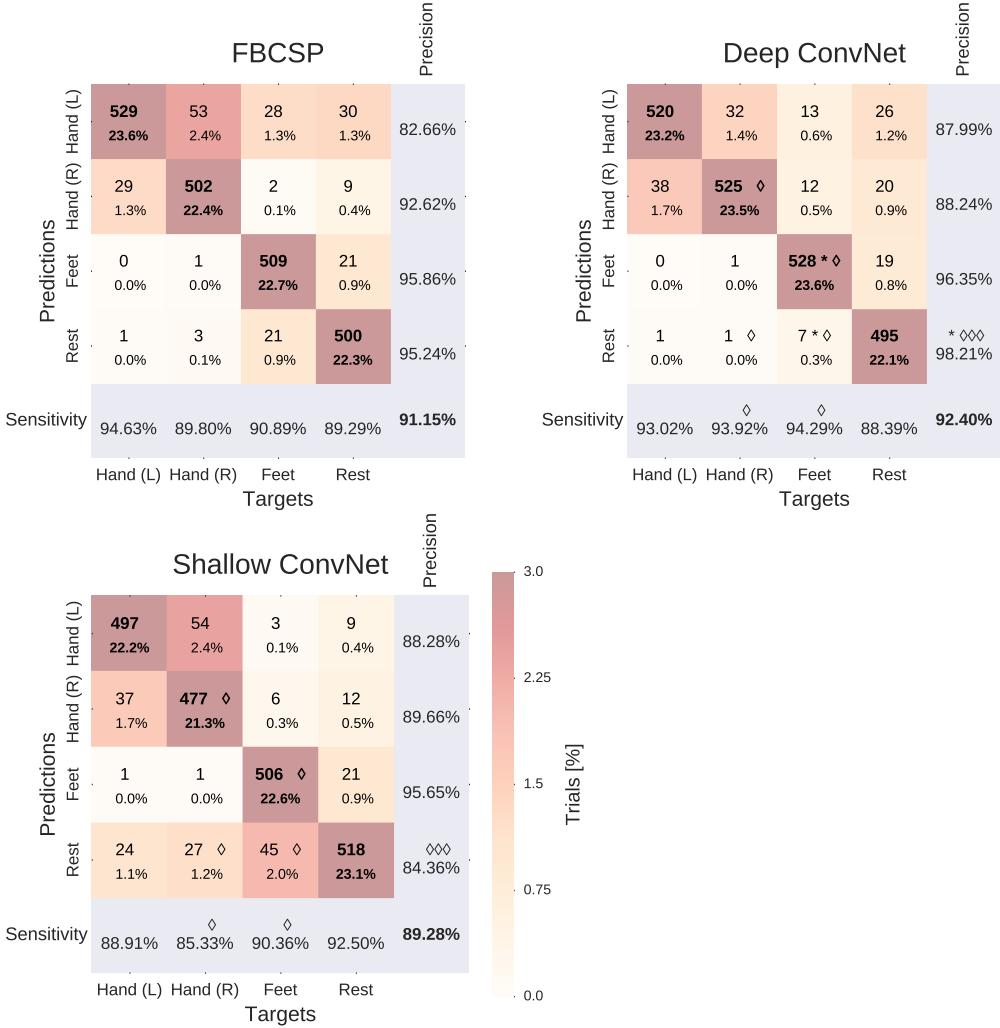
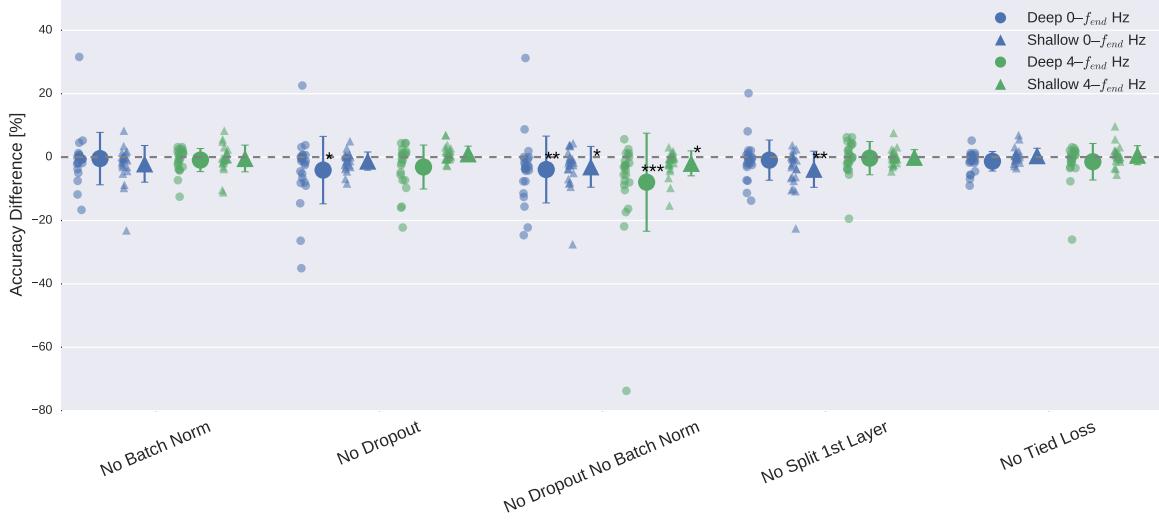
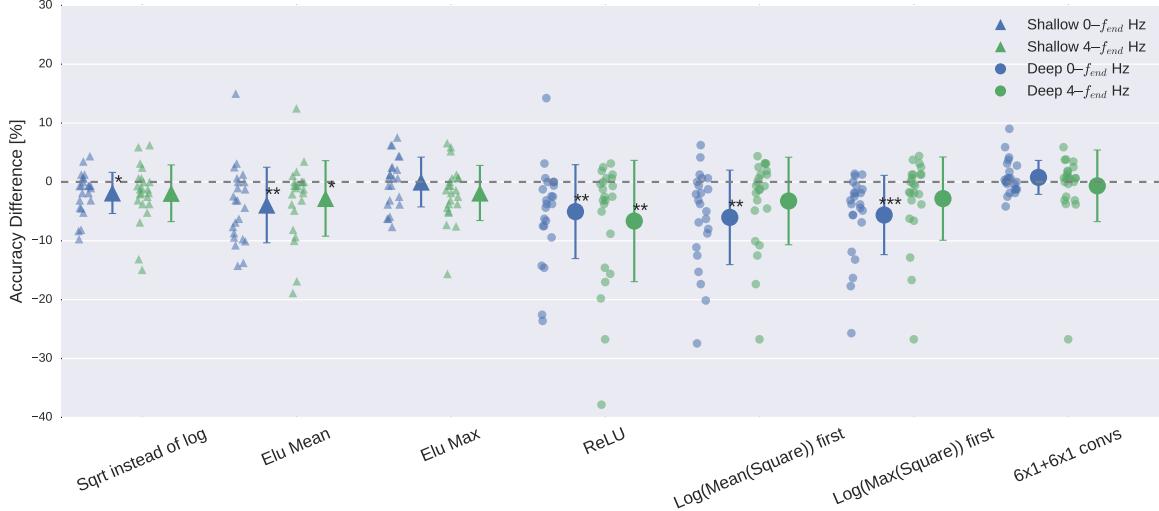


Figure 7: **Confusion matrices for FBCSP- and ConvNet-based decoding.** Results are shown for the High-Gamma Dataset, on  $0-f_{end}$  Hz. Each entry of row r and column c for upper-left 4x4-square: Number of trials of target r predicted as class c (also written in percent of all trials). Bold diagonal corresponds to correctly predicted trials of the different classes. The lower-right value corresponds to overall accuracy. Bottom row corresponds to sensitivity defined as the number of trials correctly predicted for class c / number of trials for class c. Rightmost column corresponds to precision defined as the number of trials correctly predicted for class r / number of trials predicted as class r. Stars indicate statistically significant differences of ConvNet decoding from FBCSP, diamonds indicate statistically significant differences between the shallow and deep ConvNets.  $p < 0.05$ : ♦/\*,  $p < 0.01$ : ♦♦/\*\*,  $p < 0.001$ : ♦♦♦/\*\*\*, Wilcoxon signed-rank test.



(a) Impact of design choices applicable to both ConvNets. Shown are the effects from the removal of one aspect from the architecture on decoding accuracies. All statistically significant differences were accuracy decreases. Notably, there was a clear negative effect of removing both dropout and batch normalization, seen in both ConvNets' accuracies and for both frequency ranges.



(b) Impact of different types of nonlinearities, pooling modes and filter sizes. Results are given independently for the deep ConvNet and the shallow ConvNet. As before, all statistically significant differences were from accuracy decreases. Notably, replacing ELU by ReLU as nonlinearity led to decreases on both frequency ranges, which were both statistically significant.

**Figure 8: Impact of ConvNet design choices on decoding accuracy.** Accuracy differences of baseline and design choices on x-axis for the  $0-f_{end}$ -Hz and  $4-f_{end}$ -Hz datasets. Each small marker represents accuracy difference for one subject, each larger marker represents mean accuracy difference across all subjects of both datasets. Bars: standard error of the differences across subjects. Stars indicate statistically significant differences to baseline (Wilcoxon signed-rank test,  $p < 0.05$ : \*;  $p < 0.01$ : \*\*;  $p < 0.001$ : \*\*\*)

Dataset	Frequency range [Hz]	Accuracy	Difference to deep	p-value
BCIC	0–38	67.7	-3.2	0.13
BCIC	4–38	60.8	-9.3	0.004**
HGD	0–125	88.9	-3.5	0.020*
HGD	4–125	89.8	-1.6	0.54
Combined	0– $f_{end}$	80.6	-3.4	0.004**
Combined	4– $f_{end}$	78.5	-4.9	0.01*

Table 4: **Decoding accuracies residual networks and difference to deep ConvNets.** BCIC: BCI Competition Dataset. HGD: High-Gamma Dataset. Accuracy is mean accuracy in percent. P-value from Wilcoxon signed-rank test for the statistical significance of the differences to the deep ConvNet (cropped training). Accuracies were always slightly worse than deep ConvNet, statistically significantly different for both frequency ranges on the combined dataset.

For the deep ConvNet, using ReLU instead of ELU as nonlinearity in all layers worsened performance ( $p < 0.01$ , see Figure 8b on the right side). Replacing the  $10 \times 1$  convolutions by  $6 \times 1 + 6 \times 1$  convolutions did not statistically significantly affect the performance ( $p > 0.4$ ).

#### Result 6 Recent deep learning advances substantially increased accuracies

Figure 9 clearly shows that only recent advances in deep learning methods together (by which we mean the combination of batch normalization, dropout and ELUs) allowed our deep ConvNet to be competitive with FBCSP. Without these recent advances, the deep ConvNet had statistically significantly worse accuracies than FBCSP for both  $0-f_{end}$ -Hz and  $4-f_{end}$ -Hz data ( $p < 0.001$ , Wilcoxon signed-rank test). The shallow ConvNet was less strongly affected, with no statistically significant accuracy difference to FBCSP ( $p > 0.2$ ).

#### Result 7 Residual network performed worse than deep ConvNet

Residual networks had consistently worse accuracies than the deep ConvNet as seen in Table 4. All accuracies were lower and the difference was statistically significant for both frequency ranges on the combined dataset.

### 3.3 Training Strategy

#### Result 8 Cropped training strategy improved deep ConvNet on higher frequencies

Cropped training increased accuracies statistically significantly for the deep ConvNet on the  $4-f_{end}$ -Hz data ( $p < 1e-5$ , Wilcoxon signed-rank test). In all other settings ( $0-f_{end}$ -Hz data, shallow ConvNet), the accuracy differences were not statistically significant ( $p > 0.1$ ) and showed a lot of variation between subjects.

#### Result 9 Training ConvNets took substantially longer than FBCSP

FBCSP was substantially faster to train than the ConvNets with cropped training, by a factor of 27–45 on the BCI Competition Dataset and a factor of 5–9 on the High-Gamma Dataset. Training times are end-to-end, i.e., include the loading and preprocessing of the data. These times are only

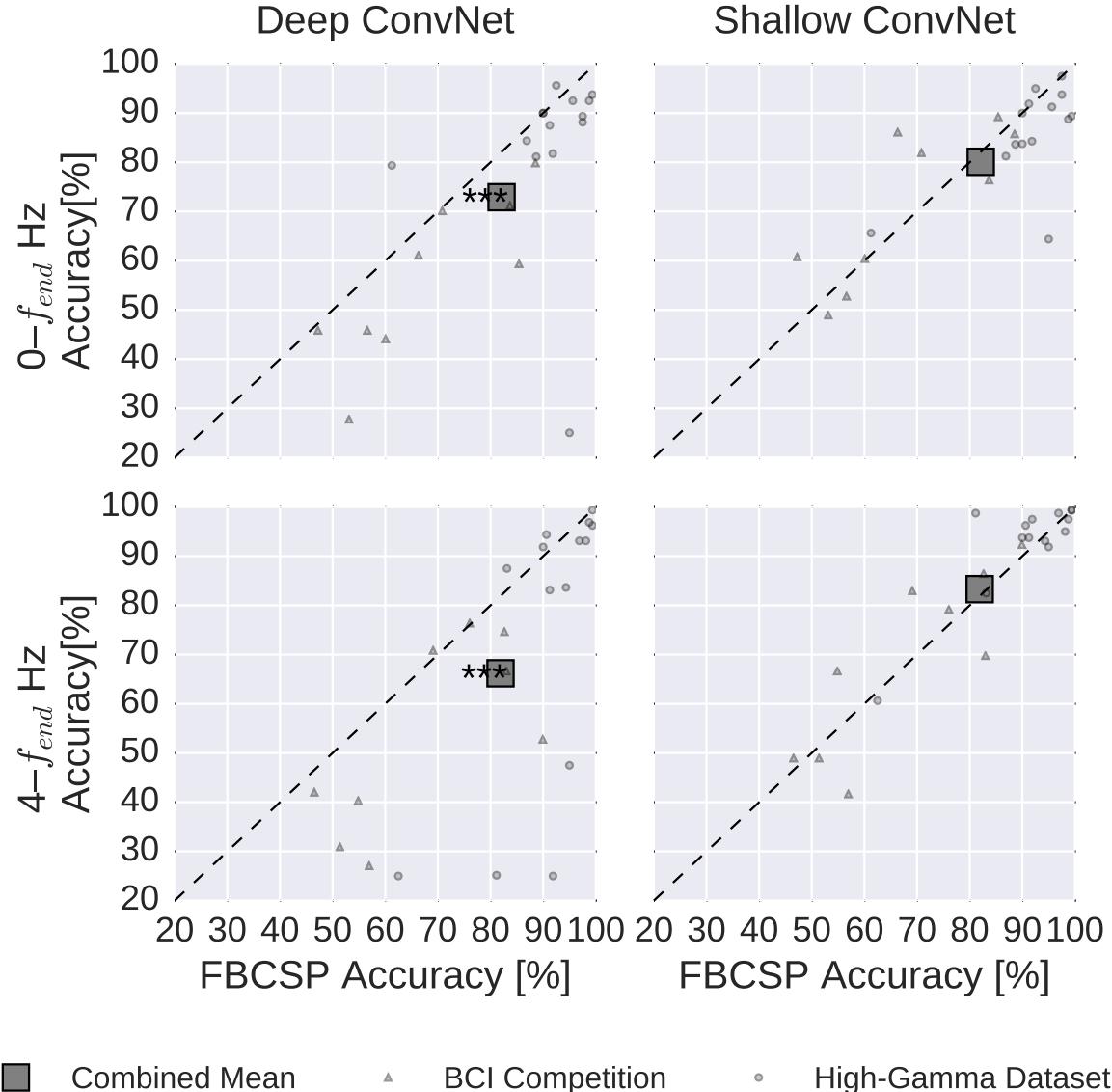


Figure 9: **Impact of recent advances on overall decoding accuracies.** Accuracies without batch normalization, dropout and ELUs. All conventions as in Figure 6. In contrast to the results on Figure 6, the deep ConvNet without implementation of these recent methodological advances performed worse than FBCSP; the difference was statistically significant for both frequency ranges.

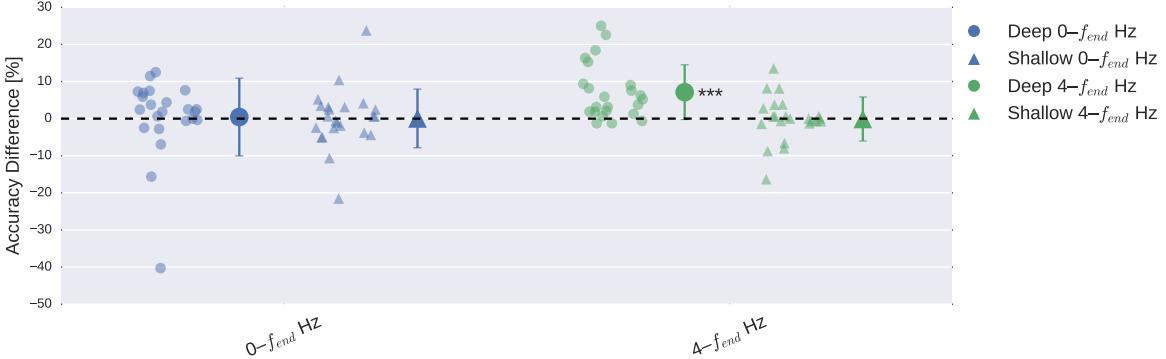


Figure 10: **Impact of training strategy (cropped vs trial-wise training) on accuracy.** Accuracy difference for both frequency ranges and both ConvNets when using cropped training instead of trial-wise training. Other conventions as in Figure 8. Cropped training led to better accuracies for almost all subjects for the deep ConvNet on the  $4-f_{end}$ -Hz frequency range.

Dataset	FBCSP	std	Deep ConvNet	std	Shallow ConvNet	std
BCIC	00:00:33	<00:00:01	00:24:46	00:06:01	00:15:07	00:02:54
HGD	00:06:40	00:00:54	1:00:40	00:27:43	00:34:25	00:16:40

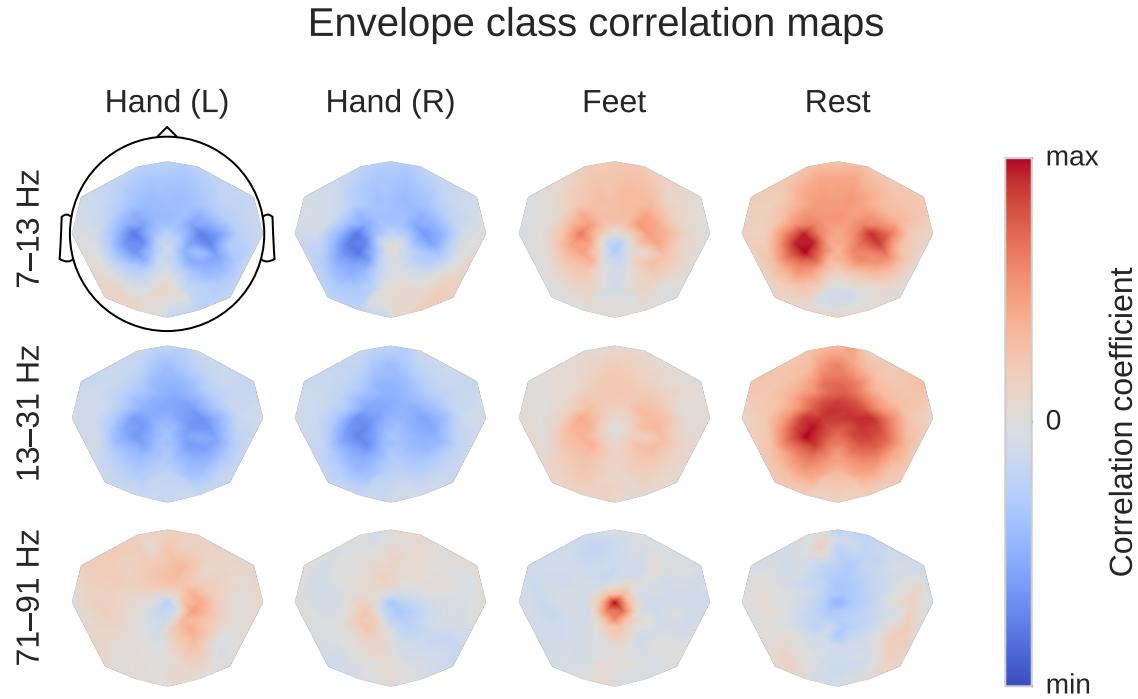
Table 5: **Training times.** Mean times across subjects given in Hours:Minutes:Seconds. BCIC: BCI Competition Dataset. HGD: High-Gamma Dataset. Std is standard deviation across subjects. ConvNets take substantially longer to train than FBCSP, especially the deep ConvNet.

meant to give a rough estimate of the training times as there were differences in the computing environment between ConvNets training and FBCSP training. Most importantly, FBCSP was trained on CPU, while the networks were trained on GPUs (see Section A.9). Longer relative training times for FBCSP on the High-Gamma Dataset can be explained by the larger number of frequency bands we use on the High-Gamma Dataset. Online application of the trained ConvNets does not suffer from the same speed disadvantage compared to FBCSP; the fast prediction speed of trained ConvNets make them well suited for decoding in real-time BCI applications.

### 3.4 Visualization

**Result 10** *Band power topographies show event-related “desynchronization/synchronization” typical for motor tasks*

Before moving to ConvNet visualization, we examined the spectral amplitude changes associated with the different movement classes in the alpha, beta and gamma frequency bands, finding the expected overall scalp topographies (see Figure 11). For example, for the alpha (7–13 Hz) frequency band, there was a class-related power decrease (anti-correlation in the class-envelope correlations) in the left and right pericentral regions with respect to the hand classes, stronger contralaterally to the side of the hand movement , i.e., the regions with pronounced power decreases lie around

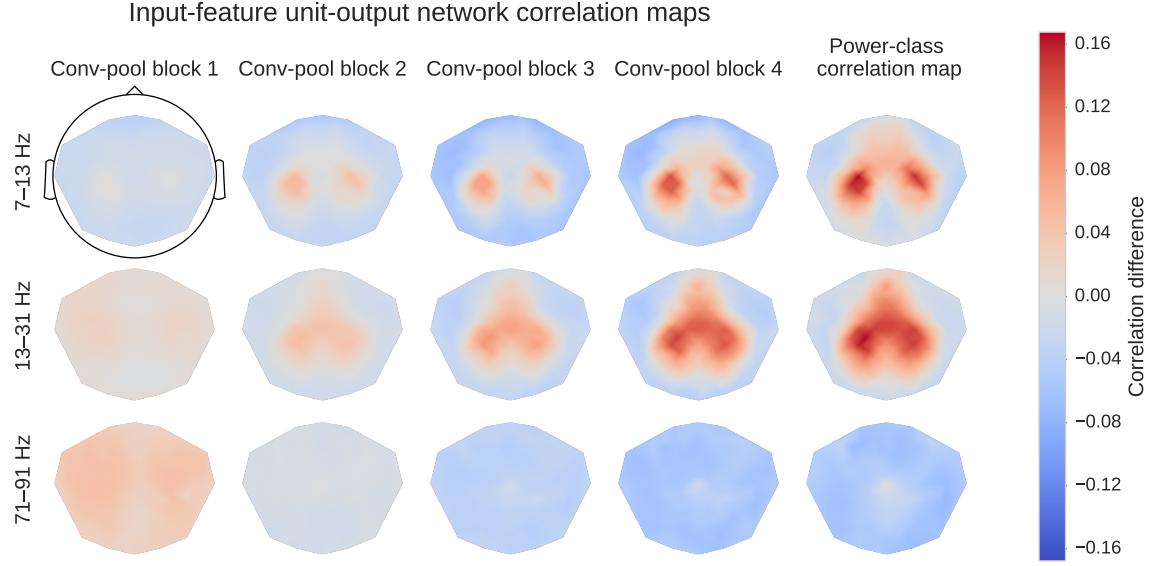


**Figure 11: Envelope-class correlations for alpha, beta and gamma bands for all classes.** Average over subjects from the High-Gamma Dataset. Colormaps are scaled per frequency band/column. This is a ConvNet-independent visualization, for an explanation of the computation see Section A.5.1. Scalp plots show spatial distributions of class-related spectral amplitude changes well in line with the literature.

the primary sensorimotor hand representation areas. For the feet class, there was a power decrease located around the vertex, i.e., approx. above the primary motor foot area. As expected, opposite changes (power increases) with a similar topography were visible for the gamma band (71–91 Hz).

**Result 11** *Input-feature unit-output correlation maps show learning progression through the ConvNet layers*

We used our input-feature unit-output correlation mapping technique to examine the question how correlations between EEG power and the behavioral classes are learnt by the network. Figure 12 shows the input-feature unit-output correlation maps for all four conv-pooling-blocks of the deep ConvNet, for the group of subjects of the High-Gamma Dataset. As a comparison, the Figure also contains the correlation between the power and the classes themselves as described in Section A.5.1. The differences of the absolute correlations show which regions were more correlated with the unit outputs of the trained ConvNet than with the unit outputs of the untrained ConvNet; these correlations are naturally undirected. Overall, the input-feature unit-output correlation maps became more similar to the power-class correlation maps with increasing layer depth. This gradual progression was also reflected in an increasing correlation of the unit outputs with the class labels



**Figure 12: Power input-feature unit-output network correlation maps for all conv-pool blocks of the deep ConvNet.** Correlation difference indicates the difference of correlation coefficients obtained with the trained and untrained model for each electrode respectively and is visualized as a topographic scalp plot. Details see Section A.5.1. Rightmost column shows the correlation between the envelope of the EEG signals in each of the three analyzed frequency bands and the four classes. Notably, the absolute values of the correlation differences became larger in the deeper layers and converged to patterns that were very similar to those obtained from the power-class correlations.

with increasing depth of the layer (see Figure 13).

**Result 12** *Input-perturbation network-prediction correlation maps show causal effect of spatially localized band power features on ConvNet predictions*

We show three visualizations extracted from input-perturbation network-prediction correlations, the first two to show the frequency profile of the causal effects, the third to show their topography.

Thus, first, we computed the mean across electrodes for each class separately to show correlations between classes and frequency bands. We see plausible results, for example, for the rest class, positive correlations in the alpha and beta bands and negative correlations in the gamma band (see Figure 14).

Then, second, by taking the mean of the absolute values both over all classes and electrodes, we computed a general frequency profile. This showed clear peaks in the alpha, beta and gamma bands (see Figure 15). Similar peaks were seen in the means of the CSP binary decoding accuracies for the same frequency range.

Thirdly, scalp maps of the input-perturbation effects on network predictions for the different frequency bands, as shown in Figure 16, show spatial distributions expected for motor tasks in the alpha, beta and — for the first time for such a non-invasive EEG decoding visualization — for the high gamma band. These scalp maps directly reflect the behavior of the ConvNets and one needs to

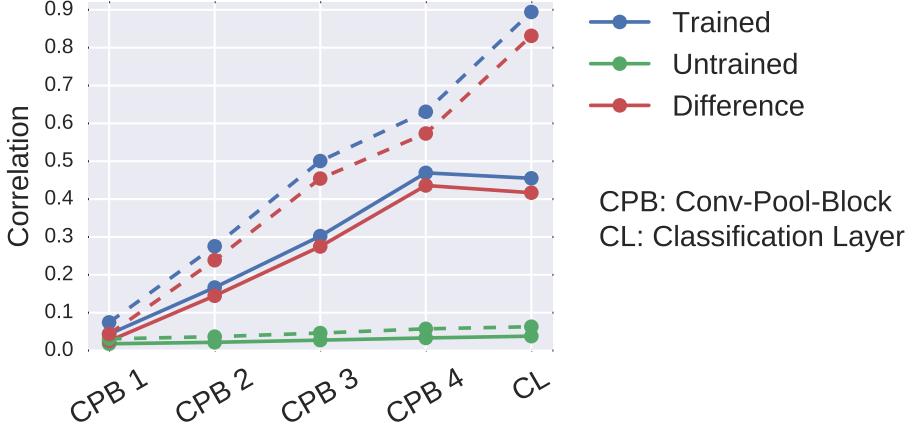


Figure 13: **Absolute correlations between unit outputs and class labels.** Each dot represents absolute correlation coefficients for one layer of the deep ConvNet. Solid lines indicate result of taking mean over absolute correlation coefficients between classes and filters. Dashed lines indicate result of first taking the maximum absolute correlation coefficient per class (maximum over filters) and then the mean over classes. Absolute correlations increased almost linearly with increasing depth of the layer.

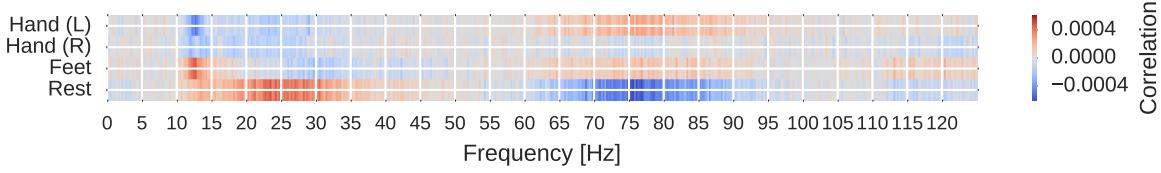


Figure 14: **Input-perturbation network-prediction correlations for all frequencies for the deep ConvNet, per class.** Plausible correlations, e.g., rest positively, other classes negatively correlated with the amplitude changes in frequency range from 20 Hz to 30 Hz.

be careful when making inferences about the data from them. For example, the positive correlation on the right side of the scalp for the Hand (R) class in the alpha band only means the ConvNet increased its prediction when the amplitude at these electrodes was increased independently of other frequency bands and electrodes. It does not imply that there was an increase of amplitude for the right hand class in the data. Rather, this correlation could be explained by the ConvNet reducing common noise between both locations, for more explanations of these effects in case of linear models see Haufe et al. (2014). Nevertheless, for the first time in non-invasive EEG, these maps clearly revealed the global somatotopic organization of causal contributions of motor cortical gamma band activity to decoding right and left hand as well as foot movements.

In summary, our visualization methods proved useful to map the spatial distribution of the features learned by the ConvNets to perform single-trial decoding of the different movement classes and in different physiologically important frequency bands.

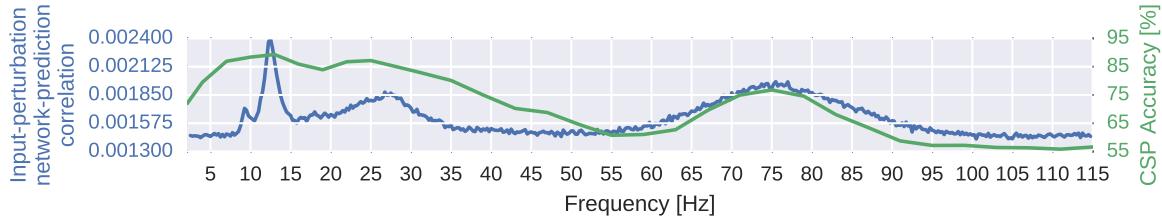


Figure 15: **Absolute input-perturbation network-prediction correlation frequency profile for the deep ConvNet.** Mean absolute correlation value across classes. CSP binary decoding accuracies for different frequency bands for comparison, averaged across subjects and class pairs. Peaks in alpha, beta and gamma band for input-perturbation network-prediction correlations and CSP accuracies.

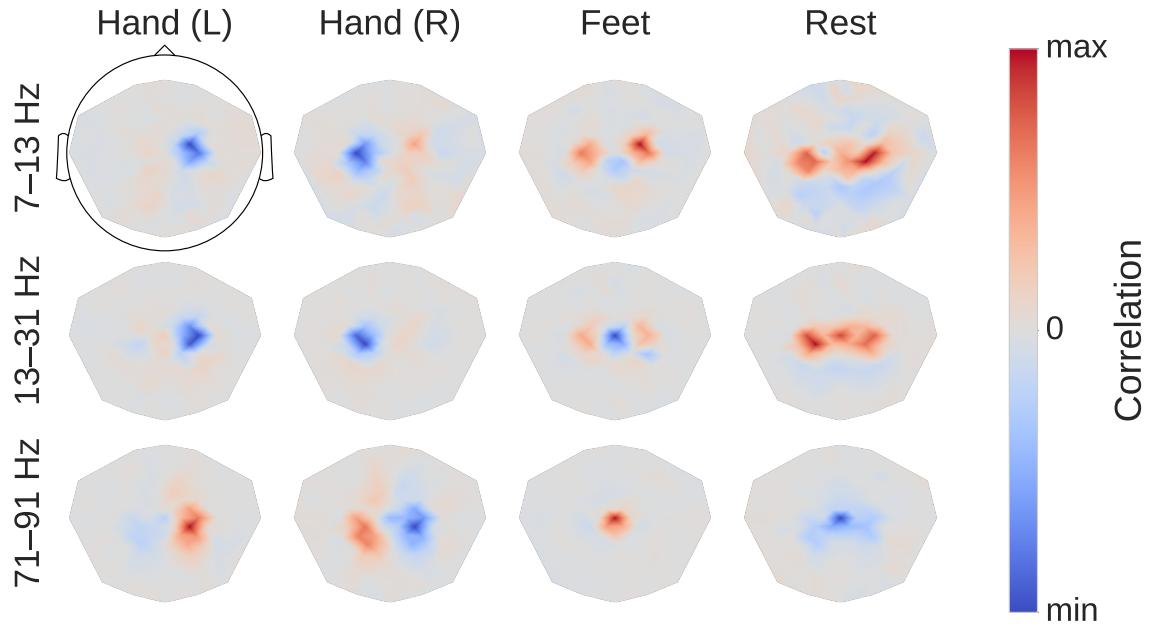


Figure 16: **Input-perturbation network-prediction correlation maps for the deep ConvNet.** Correlation of class predictions and amplitude changes. Averaged over all subjects of the High-Gamma Dataset. Colormaps are scaled per scalp plot. Plausible scalp maps for all frequency bands, e.g. contralateral positive correlations for the hand classes in the gamma band.

## 4 Discussion

This study systematically evaluated ConvNet of different architectures and with different design choices against a validated baseline method, i.e. FBCSP. This study shows that ConvNets allow accurate motor decoding from EEG, that recent deep-learning techniques are critical to boost ConvNet performance, and that a cropped ConvNet training strategy can further increase decoding performance. Thus, ConvNets can achieve successful end-to-end learning from EEG with just minimal preprocessing. This study also demonstrates that novel ConvNets visualization offer new possibilities in brain mapping of informative EEG features.

### 4.1 Architectures and design choices

#### 4.1.1 ConvNets vs. FBCSP

Our results demonstrate that deep and shallow ConvNets, with appropriate design choices, are able to — at least — reach the accuracies of FBCSP for motor decoding from EEG (see Result 2). In our main comparison for the combined datasets (see Table 2), the accuracies of both deep and shallow ConvNets are very close and slightly higher than the accuracies of FBCSP. As filter bank common spatial patterns is the de facto standard for motor decoding from EEG recordings, this strongly implies ConvNets are also a suitable method for motor decoding. While we have shown deep ConvNets to be competitive with standard FBCSP, a lot of variants of FBCSP exist. For example, many regularized variants of CSP exist that can be used inside FBCSP (Lotte and Guan, 2011; Samek, 2014); a comparison to these could further show the exact tradeoff between the more generic ConvNets and the more domain-specific FBCSP.

#### 4.1.2 Role of recent deep learning advances

Success depends on using recent developments in deep learning. The accuracy increase that we demonstrate when using batch normalization, dropout and exponential linear units implies that general advances in deep learning can also improve brain-signal decoding. The improvement from using these techniques replicates recent findings in computer vision and other fields. In our study, improvements were most pronounced for the deep ConvNet on 4– $f_{end}$ Hz data (see Result 6), indicating that the networks can easily overfit in this setting, where band power features are likely dominant. This is consistent with our observation that cropped training, which combats overfitting by increasing the number of training examples, also drastically increased accuracies on 4– $f_{end}$ Hz data (see Result 8). There seemed to be some further gains when combining both batch normalization and dropout, albeit with some variation across architectures and frequency bands. This improvement was not clear from the start as batch normalization can in some cases remove the need for dropout (Ioffe and Szegedy, 2015), however this improvement was also found in another study using ConvNets to decode EEG data (Lawhern et al., 2016). The smaller improvement batch normalization yielded for the deep ConvNet is consistent with the claim that ELUs already allow fast learning (Clevert et al., 2016). However, all of these findings are limited by the fact that there can be interactions between these methods and with all other hyperparameters. As of yet, we also do not have a clear explanation for the large difference in accuracies obtained with ReLUs compared to ELUs; a recent study on computer vision tasks did not find these differences (Mishkin et al., 2016). Mathematically and empirically analyzing the behavior of ELUs and ReLUs for oscillatory signals and typical EEG noise might shed some light on plausible reasons.

#### 4.1.3 ConvNet architectures and interactions with discriminative features

Another finding of our study was that the shallow ConvNets performed as good as the deep ConvNets, in contrast to the hybrid and residual architectures (see Results 2, 4 and 7). These observations could possibly be better understood by investigating more closely what discriminative features there are in the EEG data and what architectures can hence best use them. For example, it would be interesting to study the effect of more layers when the networks use mostly EEG band power features, phase-related features, or a combination thereof (c.f. Hammer et al. (2013), for the role of power and phase in motor decoding) and whether there are features for which a deeper hierarchical representation could be beneficial.

We observed that squaring was important for the shallow but not for the deep ConvNet (see Result 5). The worse performance of the shallow ConvNet with ELU instead of squaring may be explained as follows. Squaring naturally allows the network to more easily extract band power features: In combination with the approximately zero-mean input, the network would already capture the signal's variance by squaring. To see this, assume that the two bandpass-filter-like and spatial-filter-like convolutional layers extract an oscillatory source in a specific frequency band; the squaring and mean pooling then directly computes the variance of this source in the pool regions. With ELUs instead of squaring, the positive parts of the oscillation would remain unchanged while the negative ones would be suppressed; the mean of the pool region would still be larger for larger amplitudes of the oscillation, but less strongly so than for the square activation. The effects of ELU and squaring for the deep ConvNet are less straightforward to analyze, since the pooling regions in our deep ConvNet were much smaller than for the shallow ConvNet (3 vs 75 samples) and might thus not cover a large enough time span to compute a very robust and informative variance average.

#### 4.1.4 Possibilities for substantial decoding accuracy improvements?

In the analyses presented here, ConvNets did not improve accuracies over FBCSP by a large margin. Significant improvements, if present, were never larger than 3.5 percent on the combined dataset with a lot of variation per subject (see Result 2). However, the deep ConvNets as used here may have learned features different from FBCSP, which could explain their higher accuracies in the lower frequencies where band power features may be less important (Hammer et al., 2013). Nevertheless, ConvNets failed to clearly outperform FBCSP in our experiments. Several reasons might contribute to this: the datasets might still not be large enough to reveal the full potential of deeper convolutional networks in EEG decoding; or the class-discriminative features might not have enough hierarchical structure which deeper ConvNets could exploit. The dataset-size issue could be solved by either creating larger datasets or also by using transfer learning approaches across subjects and/or other datasets. Further analysis of the data itself and of the convolutional networks might help to shed light whether there are features with a lot of hierarchical structure. Finally, recurrent networks could exploit signal changes that happen on longer timescales, e.g., electrodes slowly losing scalp contact over the course of a session, changes of the electrode cap position or nonstationarities in the brain signals. Thus, there is clearly still a large potential for methodological improvement in ConvNet-based EEG decoding.

These methodological improvements might also come from further methodological advances in deep learning, such as newer forms of hyperparameter optimization, in case these advances also translate to even better EEG decoding accuracies. As discussed above, recent advances like dropout, batch normalization and exponential linear units can substantially improve the performance of EEG decoding with ConvNets, especially for our deep architecture. Therefore, using other recent tech-

niques, such as newer forms of hyperparameter optimization (Domhan et al., 2015; Klein et al., 2016; Springenberg et al., 2016) hold promise to further increase accuracies of ConvNets for brain-signal decoding. Furthermore, as the field is still evolving at a fast pace, new techniques can be expected to be developed and might then also benefit brain-signal decoders using convolutional neural networks.

However, methodological improvements may also happen in the broad field of “non-ConvNet” approaches. Obviously, currently no final verdict is possible about an “optimal” method for EEG decoding, if there is a single best method for the large variety of EEG decoding problems at all. The findings of the present study however support that ConvNet-based decoding is a contender in this competition.

#### 4.1.5 Further potential advantages of ConvNets for brain-signal decoding

Besides the decoding performance, there are also other potential advantages of using deep ConvNets for brain-signal decoding. First, several use cases desirable for brain-signal decoding are very easy to do with deep ConvNets iteratively trained in an end-to-end fashion: Deep ConvNets can be applied to other types of tasks such as as workload estimation, error- or event-related potential decoding (as others have started (Lawhern et al., 2016)) or even other types of recordings such as MEG or ECoG. Also, ConvNets, due to their iterative training, have a natural way of pretraining and finetuning; for example a ConvNet can be pretrained on data from the past or data from other subjects and then be finetuned with new data from a new subject. Finetuning can be as simple as continuing the iterative training process on the new data, possibly with a smaller learning rate and this finetuning can also be used to perform supervised online adaptation. Second, due to their joint optimization, single ConvNets can be building blocks for more sophisticated setups of multiple ConvNets. One recent example attempts to create ConvNets that are robust to changes in the input distribution (Ganin et al., 2016). This could be used to alleviate the long-standing EEG decoding problem of changes in the EEG signal distribution from one session to another.

## 4.2 Training strategy

### 4.2.1 Cropped training effect on accuracies

We observed that cropped training was necessary for the deep ConvNet to reach competitive accuracies on the dataset excluding very low frequencies (see Result 8). The large increase in accuracy with cropped training for the deep network on the  $4-f_{end}$ -Hz data might indicate a large number of training examples is necessary to learn to extract band power features. This makes sense as the shifted neighboring windows may contain the same, but shifted, oscillatory signals. These shifts could prevent the network from overfitting on phase information within the trial, which is less important in the higher than the lower frequencies (Hammer et al., 2013). This could also explain why other studies on ConvNets for brain-signal decoding, which did not use cropped training, but where band power might be the most discriminative feature, have used fairly shallow architectures and sometimes found them to be superior to deeper versions (Stober et al., 2014).

### 4.2.2 Suitability for online decoding

Our cropped training strategy appears particularly well-applicable for online brain-signal decoding. As described above, it may offer performance advantages compared with conventional (non-cropped) training. Additionally, cropped training allows for a useful calibration of the trade-off between

decoding delay and decoding accuracy in online settings. The duration from trial start until the last sample of the first crop should roughly correspond to the minimum time needed to decode a control signal. Hence, smaller crops can allow less delay — the first small crop could end at an early sample within the trial without containing too many timesteps from before the trial that could otherwise disturb the training process. Conversely, larger crops that still contain mostly timesteps from within the trial imply a larger delay until a control signal is decoded while possibly increasing the decoding accuracy due to more information contained in the larger crops. These intuitions should be confirmed in online experiments.

### 4.3 Visualization

#### 4.3.1 Insights from current visualizations

In addition to exploring how ConvNets can be successfully used to decode information from the EEG, we have also developed and tested two complementary methods to visualize what ConvNets learn from the EEG data. So far, the literature on using ConvNets for brain-signal decoding has, for example, visualized weights or outputs of ConvNet layers (Bashivan et al., 2016; Santana et al., 2014; Stober, 2016; Yang et al., 2015), determined inputs that maximally activate specific convolutional filters (Bashivan et al., 2016), or described attempts at synthesizing the preferred input of a convolutional filter (Bashivan et al., 2016) (see Supplementary Section A.1 for a more extensive overview). Here, we applied both a correlative and a causally interpretable visualization method to visualize the frequencies and spatial distribution of band power features used by the networks. The visualizations showed plausible spatial distributions for motor tasks in the alpha, beta and gamma bands (see Section 3.4). The input-feature unit-output and the input-perturbation network-prediction correlation maps together clearly showed that the deep ConvNet learned to extract and use band power features with specific spatial distributions. Hence, while the computation of power was built into both the FBCSP and shallow ConvNet, our deep ConvNets successfully learned to perform the computation of band power features from the raw input in an end-to-end manner. Our network correlation maps can readily show spatial distributions per subject and for the whole group of subjects. Thus, our network correlation maps proved useful as a technique for spatially mapping the features learned by the ConvNets to perform single-trial decoding.

#### 4.3.2 Feature discovery through more sophisticated visualizations?

One limitation of the visualizations presented here is that so far, we only designed them to show how ConvNets use known band power features. However, it could be even more interesting to investigate whether novel or so-far unknown features are used and to characterize them. This could be especially informative for tasks where the discriminative features are less well known than for motor decoding, e.g. for less-investigated tasks such as decoding of task performance (Meinel et al., 2016). But even for the data used in this study, our results show hints that deep ConvNets used different features than shallow ConvNets as well as the FBCSP-based decoding, since there are statistically significant differences between their confusion matrices (see Result 3). This further strengthens the motivation to explore what features the deep ConvNet exploits, for example using visualizations that show what parts of a trial are relevant for the classification decision or what a specific convolutional filter/unit output encodes. Newer visualization methods such as layer-wise relevance propagation (Bach et al., 2015; Montavon et al., 2017), inverting convolutional networks

with convolutional networks ([Dosovitskiy and Brox, 2016](#)) or synthesizing preferred inputs of units ([Nguyen et al., 2016](#)) could be promising next steps in that direction.

#### 4.4 Conclusion

In conclusion, ConvNets are not only a novel, promising tool in the EEG decoding toolbox, but combined with innovative visualization techniques, they may also open up new windows for EEG-based brain mapping.

## Acknowledgements

This work was supported by the BrainLinks-BrainTools Cluster of Excellence (DFG grant EXC1086) and by the Federal Ministry of Education and Research (BMBF, grant Motor-BIC 13GW0053D).

## Conflicts of interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

## References

- Abdel-Hamid, O., Mohamed, A. r., Jiang, H., Deng, L., Penn, G., and Yu, D. (2014). Convolutional Neural Networks for Speech Recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(10):1533–1545.
- Ang, K. K., Chin, Z. Y., Zhang, H., and Guan, C. (2008). Filter Bank Common Spatial Pattern (FBCSP) in Brain-Computer Interface. In *IEEE International Joint Conference on Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence)*, pages 2390–2397.
- Antoniades, A., Spyrou, L., Took, C. C., and Sanei, S. (2016). Deep learning for epileptic intracranial EEG data. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6.
- Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., and Samek, W. (2015). On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation. *PLOS ONE*, 10(7):e0130140.
- Ball, T., Demandt, E., Mutschler, I., Neitzel, E., Mehring, C., Vogt, K., Aertsen, A., and Schulze-Bonhage, A. (2008). Movement related activity in the high gamma range of the human EEG. *NeuroImage*, 41(2):302–310.
- Bashivan, P., Rish, I., Yeasin, M., and Codella, N. (2016). Learning Representations from EEG with Deep Recurrent-Convolutional Neural Networks. In *arXiv:1511.06448 [cs]*. arXiv: 1511.06448.
- Blankertz, B., Tomioka, R., Lemm, S., Kawanabe, M., and Muller, K.-R. (2008). Optimizing Spatial filters for Robust EEG Single-Trial Analysis. *IEEE Signal Processing Magazine*, 25(1):41–56.
- Brunner, C., Leeb, R., Müller-Putz, G., Schlögl, A., and Pfurtscheller, G. (2008). BCI Competition 2008–Graz data set A. *Institute for Knowledge Discovery (Laboratory of Brain-Computer Interfaces), Graz University of Technology*, pages 136–142.
- Buzsáki, G. and Draguhn, A. (2004). Neuronal Oscillations in Cortical Networks. *Science*, 304(5679):1926–1929.
- Canolty, R. T., Edwards, E., Dalal, S. S., Soltani, M., Nagarajan, S. S., Kirsch, H. E., Berger, M. S., Barbaro, N. M., and Knight, R. T. (2006). High gamma power is phase-locked to theta oscillations in human neocortex. *Science*, 313(5793):1626–1628.
- Cecotti, H. and Graser, A. (2011). Convolutional Neural Networks for P300 Detection with Application to Brain-Computer Interfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):433–445.
- Chatrian, G. E., Petersen, M. C., and Lazarte, J. A. (1959). The blocking of the rolandic wicket rhythm and some central changes related to movement. *Electroencephalography and Clinical Neurophysiology*, 11(3):497–510.
- Chin, Z. Y., Ang, K. K., Wang, C., Guan, C., and Zhang, H. (2009). Multi-class filter bank common spatial pattern for four-class motor imagery BCI. In *Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2009. EMBC 2009*, pages 571–574.

- Clevert, D.-A., Unterthiner, T., and Hochreiter, S. (2016). Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). In *ArXiv e-prints*, volume 1511, page arXiv:1511.07289.
- Crone, N. E., Miglioretti, D. L., Gordon, B., and Lesser, R. P. (1998). Functional mapping of human sensorimotor cortex with electrocorticographic spectral analysis. II. Event-related synchronization in the gamma band. *Brain*, 121(12):2301–2315.
- Darvas, F., Scherer, R., Ojemann, J. G., Rao, R. P., Miller, K. J., and Sorensen, L. B. (2010). High gamma mapping using EEG. *NeuroImage*, 49(1):930–938.
- Das, K., Giesbrecht, B., and Eckstein, M. P. (2010). Predicting variations of perceptual performance across individuals from neural activity using pattern classifiers. *NeuroImage*, 51(4):1425–1437.
- Dieleman, S., Heilman, M., Kelly, J., Thoma, M., Rasul, D. K., Battenberg, E., Weideman, H., Sønderby, S. K., instagibbs, Britefury, Raffel, C., Degrave, J., peterderivaz, Jon, Fauw, J. D., diogo149, Nouri, D., Schlüter, J., Maturana, D., CongLiu, Olson, E., McFee, B., and takacsg84 (2015). Lasagne: First release. DOI: 10.5281/zenodo.27878.
- Domhan, T., Springenberg, J. T., and Hutter, F. (2015). Speeding Up Automatic Hyperparameter Optimization of Deep Neural Networks by Extrapolation of Learning Curves. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.
- Dosovitskiy, A. and Brox, T. (2016). Inverting Visual Representations with Convolutional Networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv:1506.02753.
- Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.
- Gadhoumi, K., Lina, J.-M., Mormann, F., and Gotman, J. (2016). Seizure prediction for therapeutic devices: A review. *Journal of Neuroscience Methods*, 260:270–282.
- Galar, M., Fernández, A., Barrenechea, E., Bustince, H., and Herrera, F. (2011). An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. *Pattern Recognition*, 44(8):1761–1776.
- Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. (2016). Domain-Adversarial Training of Neural Networks. *Journal of Machine Learning Research*, 17(59):1–35.
- Giusti, A., Cireşan, D. C., Masci, J., Gambardella, L. M., and Schmidhuber, J. (2013). Fast image scanning with deep max-pooling convolutional neural networks. In *2013 IEEE International Conference on Image Processing*, pages 4034–4038.
- Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. <http://www.deeplearningbook.org>.
- Gratton, G. (1998). Dealing with artifacts: The EOG contamination of the event-related brain potential. *Behavior Research Methods, Instruments, & Computers*, 30(1):44–53.
- Hajinoroozi, M., Mao, Z., Jung, T.-P., Lin, C.-T., and Huang, Y. (2016). EEG-based prediction of driver's cognitive performance by deep convolutional neural network. *Signal Processing: Image Communication*, 47:549–555.

- Hammer, J., Fischer, J., Ruescher, J., Schulze-Bonhage, A., Aertsen, A., and Ball, T. (2013). The role of ECoG magnitude and phase in decoding position, velocity, and acceleration during continuous motor behavior. *Frontiers in Neuroscience*, 7:200.
- Hammer, J., Pistoohl, T., Fischer, J., Kršek, P., Tomášek, M., Marusič, P., Schulze-Bonhage, A., Aertsen, A., and Ball, T. (2016). Predominance of Movement Speed Over Direction in Neuronal Population Signals of Motor Cortex: Intracranial EEG Data and A Simple Explanatory Model. *Cerebral Cortex (New York, NY)*, 26(6):2863–2881.
- Haufe, S., Meinecke, F., Görgen, K., Dähne, S., Haynes, J.-D., Blankertz, B., and Bießmann, F. (2014). On the interpretation of weight vectors of linear models in multivariate neuroimaging. *NeuroImage*, 87:96–110.
- He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep Residual Learning for Image Recognition. *arXiv:1512.03385 [cs]*. arXiv: 1512.03385.
- Hertel, L., Barth, E., Käster, T., and Martinetz, T. (2015). Deep convolutional neural networks as generic feature extractors. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–4.
- Howard, A. G. (2013). Some Improvements on Deep Convolutional Neural Network Based Image Classification. *arXiv:1312.5402 [cs]*. arXiv: 1312.5402.
- Ioffe, S. and Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of The 32nd International Conference on Machine Learning*, pages 448–456.
- Kingma, D. P. and Ba, J. (2014). Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.
- Klein, A., Falkner, S., Bartels, S., Hennig, P., and Hutter, F. (2016). Fast Bayesian Optimization of Machine Learning Hyperparameters on Large Datasets. *arXiv:1605.07079 [cs, stat]*. arXiv: 1605.07079.
- Knops, A., Thirion, B., Hubbard, E. M., Michel, V., and Dehaene, S. (2009). Recruitment of an Area Involved in Eye Movements During Mental Arithmetic. *Science*, 324(5934):1583–1585.
- Koles, Z. J., Lazar, M. S., and Zhou, S. Z. (1990). Spatial patterns underlying population differences in the background EEG. *Brain Topography*, 2(4):275–284.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.
- Kurth-Nelson, Z., Economides, M., Dolan, R. J., and Dayan, P. (2016). Fast Sequences of Non-spatial State Representations in Humans. *Neuron*, 91(1):194–204.
- Lawhern, V. J., Solon, A. J., Waytowich, N. R., Gordon, S. M., Hung, C. P., and Lance, B. J. (2016). EEGNet: A Compact Convolutional Network for EEG-based Brain-Computer Interfaces. *arXiv:1611.08024 [cs, q-bio, stat]*. arXiv: 1611.08024.
- LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

- Ledoit, O. and Wolf, M. (2004). A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, 88(2):365–411.
- Li, X. and Wu, X. (2015). Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4520–4524.
- Liang, J., Lu, R., Zhang, C., and Wang, F. (2016). Predicting Seizures from Electroencephalography Recordings: A Knowledge Transfer Strategy. In *2016 IEEE International Conference on Healthcare Informatics (ICHI)*, pages 184–191.
- Lotte, F., Congedo, M., Lécuyer, A., Lamarche, F., and Arnaldi, B. (2007). A review of classification algorithms for EEG-based brain–computer interfaces. *Journal of Neural Engineering*, 4(2):R1.
- Lotte, F. and Guan, C. (2011). Regularizing Common Spatial Patterns to Improve BCI Designs: Unified Theory and New Algorithms. *IEEE Transactions on Biomedical Engineering*, 58(2):355–362.
- Manor, R. and Geva, A. B. (2015). Convolutional Neural Network for Multi-Category Rapid Serial Visual Presentation BCI. *Frontiers in Computational Neuroscience*, 9:146.
- Manor, R., Mishali, L., and Geva, A. B. (2016). Multimodal Neural Network for Rapid Serial Visual Presentation Brain Computer Interface. *Frontiers in Computational Neuroscience*, 10.
- Meinel, A., Castaño-Candamil, S., Reis, J., and Tangermann, M. (2016). Pre-Trial EEG-Based Single-Trial Motor Performance Prediction to Enhance Neuroergonomics for a Hand Force Task. *Frontiers in Human Neuroscience*, page 170.
- Mishkin, D., Sergievskiy, N., and Matas, J. (2016). Systematic evaluation of CNN advances on the ImageNet. *arXiv:1606.02228 [cs]*. arXiv: 1606.02228.
- Moghimi, S., Kushki, A., Guerguerian, A. M., and Chau, T. (2013). A review of EEG-based brain–computer interfaces as access pathways for individuals with severe disabilities. *Assistive technology: the official journal of RESNA*, 25(2):99–110.
- Montavon, G., Lapuschkin, S., Binder, A., Samek, W., and Müller, K.-R. (2017). Explaining non-linear classification decisions with deep Taylor decomposition. *Pattern Recognition*, 65:211–222.
- Monto, S., Palva, S., Voipio, J., and Palva, J. M. (2008). Very slow EEG fluctuations predict the dynamics of stimulus detection and oscillation amplitudes in humans. *The Journal of neuroscience*, 28(33):8268–8272.
- Münßinger, J. I., Halder, S., Kleih, S. C., Furdea, A., Raco, V., Hösle, A., and Kübler, A. (2010). Brain Painting: First Evaluation of a New Brain–Computer Interface Application with ALS-Patients and Healthy Volunteers. *Frontiers in Neuroscience*, 4.
- Nasse, F., Thurau, C., and Fink, G. A. (2009). Face detection using gpu-based convolutional neural networks. In *International Conference on Computer Analysis of Images and Patterns*, pages 83–90. Springer.
- Nguyen, A., Dosovitskiy, A., Yosinski, J., Jason, Brox, T., and Clune, J. (2016). Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. *NIPS 29*.

- Nijboer, F., Sellers, E. W., Mellinger, J., Jordan, M. A., Matuz, T., Furdea, A., Halder, S., Mochty, U., Krusienski, D. J., Vaughan, T. M., Wolpaw, J. R., Birbaumer, N., and Kübler, A. (2008). A P300-based brain-computer interface for people with amyotrophic lateral sclerosis. *Clinical Neurophysiology: Official Journal of the International Federation of Clinical Neurophysiology*, 119(8):1909–1916.
- Nunez, P. L. and Srinivasan, R. (2006). *Electric Fields of the Brain: The Neurophysics of EEG*. Oxford University Press, Oxford ; New York, 2nd ed. edition.
- Page, A., Shea, C., and Mohsenin, T. (2016). Wearable seizure detection using convolutional neural networks with transfer learning. In *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1086–1089.
- Pfurtscheller, G. (1981). Central beta rhythm during sensorimotor activities in man. *Electroencephalography and Clinical Neurophysiology*, 51(3):253–264.
- Pfurtscheller, G. and Aranibar, A. (1977). Event-related cortical desynchronization detected by power measurements of scalp EEG. *Electroencephalography and Clinical Neurophysiology*, 42(6):817–826.
- Pfurtscheller, G. and Aranibar, A. (1978). Occipital rhythmic activity within the alpha band during conditioned externally paced movement. *Electroencephalography and Clinical Neurophysiology*, 45(2):226–235.
- Pfurtscheller, G. and Aranibar, A. (1979). Evaluation of event-related desynchronization (ERD) preceding and following voluntary self-paced movement. *Electroencephalography and Clinical Neurophysiology*, 46(2):138–146.
- Pfurtscheller, G. and Berghold, A. (1989). Patterns of cortical activation during planning of voluntary movement. *Electroencephalography and Clinical Neurophysiology*, 72(3):250–258.
- Pfurtscheller, G., Flotzinger, D., and Neuper, C. (1994). Differentiation between finger, toe and tongue movement in man based on 40 Hz EEG. *Electroencephalography and Clinical Neurophysiology*, 90(6):456–460.
- Quandt, F., Reichert, C., Hinrichs, H., Heinze, H. J., Knight, R. T., and Rieger, J. W. (2012). Single trial discrimination of individual finger movements on one hand: A combined MEG and EEG study. *NeuroImage*, 59(4):3316–3324.
- Ramos-Murguialday, A., Broetz, D., Rea, M., Läer, L., Yilmaz, O., Brasil, F. L., Liberati, G., Curado, M. R., Garcia-Cossio, E., Vyziotis, A., Cho, W., Agostini, M., Soares, E., Soekadar, S., Caria, A., Cohen, L. G., and Birbaumer, N. (2013). Brain-machine interface in chronic stroke rehabilitation: A controlled study. *Annals of Neurology*, 74(1):100–108.
- Ramoser, H., Muller-Gerking, J., and Pfurtscheller, G. (2000). Optimal spatial filtering of single trial EEG during imagined hand movement. *IEEE Transactions on Rehabilitation Engineering*, 8(4):441–446.
- Ren, Y. and Wu, Y. (2014). Convolutional deep belief networks for feature extraction of EEG signal. In *2014 International Joint Conference on Neural Networks (IJCNN)*, pages 2850–2853.

- Rivet, B., Souloumiac, A., Attina, V., and Gibert, G. (2009). xDAWN Algorithm to Enhance Evoked Potentials: Application to Brain-Computer Interface. *IEEE Transactions on Biomedical Engineering*, 56(8):2035–2043.
- Sainath, T. N., Kingsbury, B., Saon, G., Soltau, H., Mohamed, A.-r., Dahl, G., and Ramabhadran, B. (2015a). Deep Convolutional Neural Networks for Large-scale Speech Tasks. *Neural Networks*, 64:39–48.
- Sainath, T. N., Vinyals, O., Senior, A., and Sak, H. (2015b). Convolutional, Long Short-Term Memory, fully connected Deep Neural Networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4580–4584.
- Sak, H., Senior, A., Rao, K., and Beaufays, F. (2015). Fast and Accurate Recurrent Neural Network Acoustic Models for Speech Recognition. In *arXiv:1507.06947 [cs, stat]*. arXiv: 1507.06947.
- Sakhavi, S., Guan, C., and Yan, S. (2015). Parallel convolutional-linear neural network for motor imagery classification. In *Signal Processing Conference (EUSIPCO), 2015 23rd European*, pages 2736–2740.
- Samek, W. (2014). *On robust spatial filtering of EEG in nonstationary environments*. PhD thesis, Berlin, Technische Universität Berlin, Diss., 2014.
- Santana, E., Brockmeier, A., and Principe, J. (2014). Joint optimization of algorithmic suites for EEG analysis. In *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 2997–3000.
- Schack, B., Vath, N., Petsche, H., Geissler, H.-G., and Möller, E. (2002). Phase-coupling of theta–gamma EEG rhythms during short-term memory processing. *International Journal of Psychophysiology*, 44(2):143–163.
- Schalk, G., McFarland, D. J., Hinterberger, T., Birbaumer, N., and Wolpaw, J. R. (2004). BCI2000: a general-purpose brain-computer interface (BCI) system. *IEEE Transactions on Biomedical Engineering*, 51(6):1034–1043.
- Schmidhuber, J. (2015). Deep Learning in Neural Networks: An Overview. *Neural Networks*, 61:85–117. arXiv: 1404.7828.
- Sercu, T., Puhrsch, C., Kingsbury, B., and LeCun, Y. (2016). Very deep multilingual convolutional neural networks for LVCSR. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4955–4959.
- Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. (2014). OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. In *International Conference on Learning Representations (ICLR 2014)*. arXiv preprint arXiv:1312.6229.
- Shamwell, J., Lee, H., Kwon, H., Marathe, A. R., Lawhern, V., and Nothwang, W. (2016). Single-trial EEG RSVP classification using convolutional neural networks. In George, T., Dutta, A. K., and Islam, M. S., editors, *SPIE Defense + Security*, volume 9836. International Society for Optics and Photonics.

- Shelhamer, E., Long, J., and Darrell, T. (2016). Fully Convolutional Networks for Semantic Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP(99):1–1.
- Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489.
- Springenberg, J. T., Klein, A., Falkner, S., and Hutter, F. (2016). Bayesian Optimization with Robust Bayesian Neural Networks. In *Advances in Neural Information Processing Systems*, pages 4134–4142.
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.
- Stansbury, D. E., Naselaris, T., and Gallant, J. L. (2013). Natural Scene Statistics Account for the Representation of Scene Categories in Human Visual Cortex. *Neuron*, 79(5):1025–1034.
- Stober, S. (2016). Learning Discriminative Features from Electroencephalography Recordings by Encoding Similarity Constraints. In *Bernstein Conference 2016*.
- Stober, S., Cameron, D. J., and Grahn, J. A. (2014). Using Convolutional Neural Networks to Recognize Rhythm Stimuli from Electroencephalography Recordings. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, NIPS’14, pages 1449–1457, Cambridge, MA, USA. MIT Press.
- Sturm, I., Lapuschkin, S., Samek, W., and Müller, K.-R. (2016). Interpretable deep neural networks for single-trial EEG classification. *Journal of Neuroscience Methods*, 274:141–145.
- Sun, G., Hu, J., and Wu, G. (2010). A novel frequency band selection method for Common Spatial Pattern in Motor Imagery based Brain Computer Interface.
- Sun, X., Qian, C., Chen, Z., Wu, Z., Luo, B., and Pan, G. (2016). Remembered or Forgotten?—An EEG-Based Computational Prediction Approach. *PLOS ONE*, 11(12):e0167497.
- Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2015). Rethinking the inception architecture for computer vision. *arXiv preprint arXiv:1512.00567*.
- Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2014). Intriguing properties of neural networks. In *International Conference on Learning Representations*.
- Tabar, Y. R. and Halici, U. (2017). A novel deep learning approach for classification of EEG motor imagery signals. *Journal of Neural Engineering*, 14(1):016003.
- Tang, Z., Li, C., and Sun, S. (2017). Single-trial EEG classification of motor imagery using deep convolutional neural networks. *Optik - International Journal for Light and Electron Optics*, 130:11–18.

- Tangermann, M., Müller, K.-R., Aertsen, A., Birbaumer, N., Braun, C., Brunner, C., Leeb, R., Mehring, C., Miller, K. J., Müller-Putz, G. R., Nolte, G., Pfurtscheller, G., Preissl, H., Schalk, G., Schlögl, A., Vidaurre, C., Waldert, S., and Blankertz, B. (2012). Review of the BCI Competition IV. *Frontiers in Neuroscience*, 6.
- Thodoroff, P., Pineau, J., and Lim, A. (2016). Learning Robust Features using Deep Learning for Automatic Seizure Detection. In *JMLR Workshop and Conference Proceedings*, volume 56.
- Tonin, L., Carlson, T., Leeb, R., and Millán, J. d. R. (2011). Brain-controlled telepresence robot by motor-disabled people. In *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 4227–4230.
- Toro, C., Deuschl, G., Thatcher, R., Sato, S., Kufta, C., and Hallett, M. (1994). Event-related desynchronization and movement-related cortical potentials on the ECoG and EEG. *Electroencephalography and Clinical Neurophysiology*, 93(5):380–389.
- Vanhatalo, S., Palva, J. M., Holmes, M. D., Miller, J. W., Voipio, J., and Kaila, K. (2004). Infraslow oscillations modulate excitability and interictal epileptic activity in the human cortex during sleep. *Proceedings of the National Academy of Sciences of the United States of America*, 101(14):5053–5057.
- Venthur, B., Dähne, S., Höhne, J., Heller, H., and Blankertz, B. (2015). Wyrm: A Brain-Computer Interface Toolbox in Python. *Neuroinformatics*, pages 1–16.
- Wang, Z., Lyu, S., Schalk, G., and Ji, Q. (2013). Deep Feature Learning Using Target Priors with Applications in ECoG Signal Decoding for BCI. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI ’13, pages 1785–1791, Beijing, China. AAAI Press.
- Wilcoxon, F. (1945). Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 1(6):80–83.
- Yang, H., Sakhavi, S., Ang, K. K., and Guan, C. (2015). On the use of convolutional neural networks and augmented CSP features for multi-class motor imagery of EEG signals classification. *Conference proceedings: ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual Conference*, 2015:2620–2623.
- Yeager, L. (2016). Effective Visualizations for Training and Evaluating Deep Models.

## A Supplementary Materials

### A.1 Related Work

Study	Decoding problem	Input do-main	Conv/ dense lay-ers	Design choices	Training strategies	External baseline	Visualization type(s)	Visualization findings
This manuscript, Schirrmeyer et. al (2017)	Imagined and executed movement classes, within subject	Time, 0–125 Hz	5/1	Different Conv-Net architectures linearities and pooling modes ularization and intermediate normalization layers torized convolutions ted vs one-step convolutions	Trial-wise vs. cropped training strategy	FBCSP + rLDA	Feature activation correlation perturbation prediction correlation	See Section 3.4 Non-Reg- Fac- Split-
Single-trial EEG classification of motor imagery using deep convolutional neural networks, Tang et al. (2017)	Imagined movement classes, within-subject	Time, 8–30 Hz	2/2					
EEGNet: A Compact Convolutional Network for EEG-based Brain-Computer Interfaces, Lawhern et al. (2016)	Oddball response (RSVP), error response (ERN), movement classes (voluntarily started and imagined)	Time, 0.1–40 Hz	3/1	Kernel sizes				
Remembered or Forgotten? — An EEG-Based Computational Prediction Approach, Sun et al (2016)	Memory performance, within-subject	Time, 0.05–15 Hz	2/2		Different time windows	Weights (spatial)	Largest weights found over prefrontal and temporal cortex	
Multimodal Neural Network for Rapid Serial Visual Presentation Brain Computer Interface, Manor et al. (2016)	Oddball response using RSVP and image (combined image-EEG decoding), within-subject	Time, 0.3–20 Hz	3/2			Weights tivations maps by gradient	Weights showed typical P300 distribution tivations were high at plausible times (300–500ms) maps showed plausible spatio-temporal plots	

Study	Decoding problem	Input do-main	Conv/ dense lay-ers	Design choices	Training strategies	External baseline	Visualization type(s)	Visualization findings
A novel deep learning approach for classification of EEG motor imagery signals, <a href="#">Tabar and Halici (2017)</a>	Imagined and executed movement classes, within-subject	Frequency, 1/1 6–30 Hz		Addition of six-layer stacked autoencoder on ConvNet features nel sizes	FBCSP, Twin SVM, DDFBS, Bi- spectrum, RQNN	Weights (spa- tial + frequen- tial)	Some weights represented difference of values of two <b>Herr</b> trodes on different sides of head	
Predicting Seizures from Electroencephalography Recordings: A Knowledge Transfer Strategy, <a href="#">Liang et al. (2016)</a>	Seizure prediction, within-subject	Frequency, 1/2 0–200 Hz		Different divisions of frequency range ferent lengths of time crops fer learning with auxiliary non-epilepsy datasets		Weights tering weights	Clusters of weights showed typical frequency band subdivision (delta, theta, alpha, beta, gamma) Dif-	Trans-
EEG-based prediction of driver's cognitive performance by deep convolutional neural network, <a href="#">Hajinorooyi et al. (2016)</a>	Driver performance, within- and cross-subject	Time, 1–50 Hz	1/3	Replacement of convolutional layers by restricted Boltzmann machines with slightly varied network architecture				
Deep learning for epileptic intracranial EEG data, <a href="#">Antoniades et al. (2016)</a>	Epileptic discharges, cross-subject	Time, 0–100 Hz	1–2/2	1 or 2 convolutional layers		Weights relation weights and interictal epileptic discharges (IED) tivations	Weights increasingly correlated with IED waveforms with increasing number of training iterations ond layer captured more complex and well-defined epileptic shapes than first layer led to highly synchronized activations for neighbouring electrodes	
Learning Robust Features using Deep Learning for Automatic Seizure Detection, <a href="#">Thodoroff et al. (2016)</a>	Start of epileptic seizure, within- and cross-subject	Frequency, 3/1 (+ LSTM mean as postprocessor) amplitude for 0–7 Hz, 7–14 Hz, 14–49 Hz			Hand crafted features + SVM	Input occlusion and effect on prediction accuracy	Allowed to locate areas critical for seizure	
Single-trial EEG RSVP classification using convolutional neural networks, <a href="#">Shamwell et al. (2016)</a>	Oddball re- sponse (RSVP), groupwise (ConvNet trained on all subjects)	Time, 0.5–50 Hz	4/3			Weights (spa- tial)	Some filter weights had expected topographic distributions for P300 ers filters had large weights on areas not traditionally associated with P300	

Study	Decoding problem	Input do-main	Conv/ dense lay-ers	Design choices	Training strategies	External baseline	Visualization type(s)	Visualization findings
Wearable seizure detection using convolutional neural networks with transfer learning, <a href="#">Page et al. (2016)</a>	Seizure detection, cross-subject, within-subject, group-wise	Time, 0–128 Hz	1-3/1-3		Cross-subject supervised training, within-subject finetuning of fully connected layers	Multiple: spectral features, higher order statistics + linear-SVM, RBF-SVM, ...		
Learning Representations from EEG with Deep Recurrent-Convolutional Neural Networks, <a href="#">Bashivan et al. (2016)</a>	Cognitive load (number of characters to memorize), cross-subject	Frequency, mean power for 4–7 Hz, 8–13 Hz, 13–30 Hz	3–7/2 (+ other temporal post-processing (see design choices))	Number of convolutional layers Temporal processing of ConvNet output by max pooling, temporal convolution, LSTM or temporal convolution + LSTM		Inputs that maximally activate given filter	Different filters were sensitive to different frequency bands	
Deep Feature Learning for EEG Recordings, <a href="#">Stober (2016)</a>	Type of music rhythm, group-wise (ensembles of leave-one-subject-out trained models, evaluated on separate test set of same subjects)	Time, 0.5–30Hz	2/1	Kernel sizes	Pretraining first layer as convolutional autoencoder with different constraints	Activations of these inputs	Layers had more spatially localized activations	“convolution” for these inputs features had noticeable links to well-known electrophysiological markers of cognitive load
Convolutional Neural Network for Multi-Category Rapid Serial Visual Presentation BCI, <a href="#">Manor and Geva (2015)</a>	Oddball response (RSVP), within-subject	Time, 0.1–50 Hz	3/3 (Spatiotemporal regularization)			Weights (spatial+3 timesteps, pretrained as autoencoder)	Different constraints led to different weights, one type of constraints could enforce weights that are similar across subjects; other type of constraints led to weights that have similar spatial topographies under different architectural configurations and preprocessings	
Parallel Convolutional-Linear Neural Network for Motor Imagery Classification, <a href="#">Sakhavi et al. (2015)</a>	Imagined movement classes, within-subject	Frequency, 4–40 Hz, using FBCSP	2/2 (Final fully connected layer uses concatenated output by convolutional and fully connected layers)	Combination ConvNet and MLP (trained on different features) vs. only ConvNet vs. only MLP	Weights and single-trial activations	Spatiotemporal regularization led to softer peaks in weights	Trial weights showed typical P300 distribution	Activations mostly had peaks at typical times (300–400ms)

Study	Decoding problem	Input do-main	Conv/ dense layers	Design choices	Training strategies	External baseline	Visualization type(s)	Visualization findings
Using Convolutional Neural networks to Recognize Rhythm Stimuli from Electroencephalography Recordings, <a href="#">Stober et al. (2014)</a>	Type of music rhythm, within-subject	Time and frequency evaluated, 0-200 Hz	1-2/1	Best values from automatic hyperparameter optimization: frequency cutoff, one vs two layers, kernel sizes, number of channels, pooling width	Best values from automatic hyperparameter optimization: learning rate, learning rate decay, momentum, final momentum			
Convolutional deep belief networks for feature extraction of EEG signal, <a href="#">Ren and Wu (2014)</a>	Imagined movement classes, within-subject	Frequency, 8-30 Hz	2/0 (Convolutional deep belief network, separately trained RBF-SVM classifier)					
Deep feature learning using target priors with applications in ECoG signal decoding for BCI, <a href="#">Wang et al. (2013)</a>	Finger flexion trajectory (regression), within-subject	Time, 0.15–200 Hz	3/1 (Convolutional layers trained as convolutional stacked autoencoder with target prior)	Partially supervised CSA				
Convolutional neural networks for P300 detection with application to brain-computer interfaces, <a href="#">Cecotti and Graser (2011)</a>	Oddball / attention response using P300 speller, within-subject	Time, 0.1-20 Hz	2/2	Electrode subset (fixed or automatically determined) using only one spatial filter	Multiple: Linear SVM, gradient boosting, E-SVM, S-SVM, mLVQ, LDA, ...	Weights	Spatial filters were similar for different architectures that filters were different (more focal, more diffuse) for different subjects Dif-	

**Table S1: Related previous publications using convolutional neural networks for EEG decoding.** Frequency domain input always only contains amplitudes or a transformation of amplitudes (power, log power, etc.), never phase information. The number of dense layers includes parametrized classification layers. Layer numbers always refer to EEG decoding models in cases of articles that use multiple modalities for decoding. Special features of the model written in parentheses after the number of layers, especially if these features make the number of layers misleading. External baseline: the study includes directly comparable baseline accuracies of non-deep-learning approaches of other authors. Visualization types and findings both refer to visualizations of the trained networks used for EEG decoding; findings are paraphrased from the original publications. Note that none of the previous studies using time-domain input showed, using a suitable visualization technique, that the ConvNets learned to use band power features, in contrast to our present study. Also note that other previous studies used artificial neural networks without convolutions for EEG analysis, e.g. [Santana et al. \(2014\)](#); [Sturm et al. \(2016\)](#).

## A.2 FBCSP implementation

As in many previous studies (Lotte et al., 2007), we used regularized linear discriminant analysis (RLDA) as the classifier, with shrinkage regularization (Ledoit and Wolf, 2004). To decode multiple classes, we used one-vs-one majority weighted voting: We trained an RLDA classifier for each pair of classes, summed the classifier outputs (scaled to be in the same range) across classes and picked the class with the highest sum (Chin et al., 2009; Galar et al., 2011).

FBCSP is typically used with feature selection, since few spatial filters from few frequency bands often suffice to reach good accuracies and using many or even all spatial filters often leads to overfitting (Blankertz et al., 2008; Chin et al., 2009). We use a classical measure for preselecting spatial filters, the ratio of the corresponding power features for both classes extracted by each spatial filter (Blankertz et al., 2008). Additionally, we performed a feature selection step on the final filter bank features by selecting features using an inner cross validation on the training set, see published code <sup>1</sup> for details.

In the present study, we designed two filter banks adapted for the two datasets to capture most discriminative motor-related band power information. In preliminary experiments, overlapping frequency bands led to higher accuracies, as also proposed by Sun et al. (2010). As the bandwidth of physiological EEG power modulations typically increases in higher frequency ranges (Buzsáki and Draguhn, 2004), we used frequency bands with 6 Hz width and 3 Hz overlap in frequencies up to 13 Hz, and bands of 8 Hz width and 4 Hz overlap in the range above 10 Hz. Frequencies above 38 Hz only improved accuracies on one of our datasets, the so-called High-Gamma Dataset (see Section 2.7, where we also describe the likely reason for this difference, namely that the recording procedure for the High-Gamma Dataset — but not for the BCI Competition Dataset — was specifically optimized for the high frequency range). Hence the upper limit of used frequencies was set at 38 Hz for the BCI Competition Dataset, while the upper limit for the High-Gamma Dataset was set to 122 Hz, close to the Nyquist frequency, thus allowing FBCSP to also use information from the gamma band.

As a sanity check, we compared the accuracies of our FBCSP implementation to those published in the literature for the same BCI Competition Dataset (Sakhavi et al., 2015), showing very similar performance: 67.59% for our implementation vs 67.01% for their implementation on average across subjects ( $p > 0.7$ , Wilcoxon signed-rank test, see Result 1 for more detailed results). This underlines that our FBCSP implementation, including our feature selection and filter bank design, indeed was a suitable baseline for the evaluation of our ConvNet decoding accuracies.

## A.3 Residual network architecture

In total, the ResNet has 31 convolutional layers, a depth where ConvNets without residual blocks started to show problems converging in the original ResNet paper (He et al., 2015). In layers where the number of channels is increased, we padded the incoming feature map with zeros to match the new channel dimensionality for the shortcut, as in option A of the original paper (He et al., 2015).

---

<sup>1</sup><https://github.com/robintibor/braindecode/blob/f9497f96a6dfdea1e24a4709a9ceb30e0f4768e3/braindecode/csp/pipeline.py#L200-L408>

Layer/Block	Number of Kernels	Kernel Size	Output Size
Input			1000x44x1
Convolution (linear)	48	3x1	1000x44x48
Convolution (ELU)	48	1x44	1000x1x48
ResBlock (ELU)	48	3x1	
ResBlock (ELU)	48	3x1	
ResBlock (ELU)	96	3x1 (Stride 2x1)	500x1x96
ResBlock (ELU)	96	3x1	
ResBlock (ELU)	144	3x1 (Stride 2x1)	250x1x96
ResBlock (ELU)	144	3x1	
ResBlock (ELU)	144	3x1 (Stride 2x1)	125x1x96
ResBlock (ELU)	144	3x1	
ResBlock (ELU)	144	3x1 (Stride 2x1)	63x1x96
ResBlock (ELU)	144	3x1	
ResBlock (ELU)	144	3x1 (Stride 2x1)	32x1x96
ResBlock (ELU)	144	3x1	
ResBlock (ELU)	144	3x1 (Stride 2x1)	16x1x96
ResBlock (ELU)	144	3x1	
Mean Pooling		10x1	7x1x144
Convolution + Softmax	4	1x1	7x1x4

Table S2: **Residual network architecture hyperparameters.** Number of kernels, kernel and output size for all subparts of the network. Output size is always time x height x channels. Note that channels here refers to input channels of a network layer, not to EEG channels; EEG channels are in the height dimension. Output size is only shown if it changes from the previous block. Second convolution and all residual blocks used ELU nonlinearities. Note that in the end we had seven outputs, i.e., predictions for the four classes, in the time dimension ( 7x1x4 final output size). In practice, when using cropped training as explained in Section 2.5.4, we even had 424 predictions, and used the mean of these to predict the trial.

#### A.4 Optimization and early stopping

Adam is a variant of stochastic gradient descent designed to work well with high-dimensional parameters, which makes it suitable for optimizing the large number of parameters of a ConvNet (Kingma and Ba, 2014). The early stopping strategy that we use throughout this study, developed in the computer vision field <sup>2</sup>, splits the training set into a training and validation fold and stops the first phase of the training when validation accuracy does not improve for a predefined number of epochs. The training continues on the combined training and validation fold starting from the parameter values that led to the best accuracies on the validation fold so far. The training ends when the loss function on the validation fold drops to the same value as the loss function on the training fold at the end of the first training phase (we do not continue training in a third phase as in the original description). Early stopping in general allows training on different types of networks and datasets without choosing the number of training epochs by hand. Our specific strategy uses the entire training data while only training once. In our study, all reported accuracies have been determined on an independent test set.

<sup>2</sup><https://web.archive.org/web/20160809230156/https://code.google.com/p/cuda-convnet/wiki/Methodology>

## A.5 Visualization methods

### A.5.1 Input-feature unit-output correlation maps

The input-feature unit-output correlation maps visualize the frequency-resolved correlation between unit outputs of the convolutional filters of the ConvNets and the power of all the samples in the receptive field of these units (see Figure S1).

To achieve this, we performed the following steps:

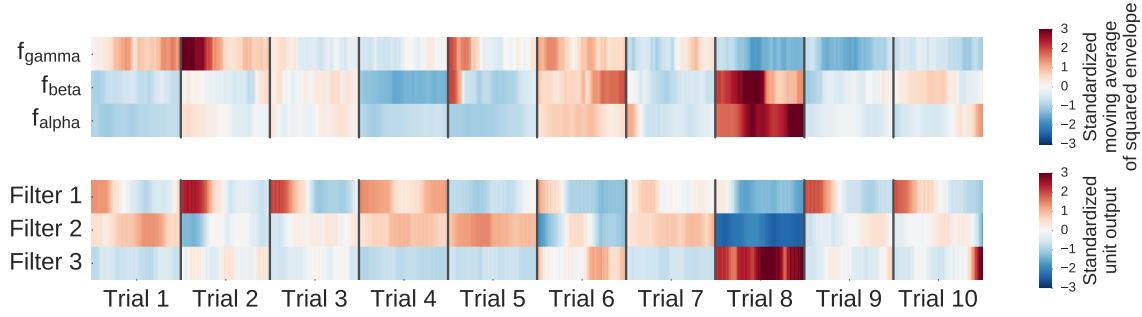
1. For each frequency band of interest, the signal was bandpass-filtered to that frequency band and the envelope was computed.
2. For each frequency band of interest, the squared mean envelope for each receptive field of a given layer was computed. We did this by computing a moving window average of the squared envelope with the moving window size the same as the receptive field size (this was the input feature for which we then evaluated how much it affected the unit output).
3. Unit outputs of the given layer on the original signal were computed.
4. Linear correlations between the squared mean envelope values for all the frequency bands and the unit outputs for each convolutional filter were computed. These correlations should reflect whether a filter might compute an approximation of the squared mean envelope of all the samples in its receptive field.

Since we compute correlations after concatenating all samples of all trials, these correlations reflect both within-trial and between-trial effects. The proposed method could, however, be straightforwardly extended to disentangle these two sources. We computed the correlations for a filter bank ranging from 0 to 119 Hz. An example result for a single electrode and a single subject is shown in Figure S2.

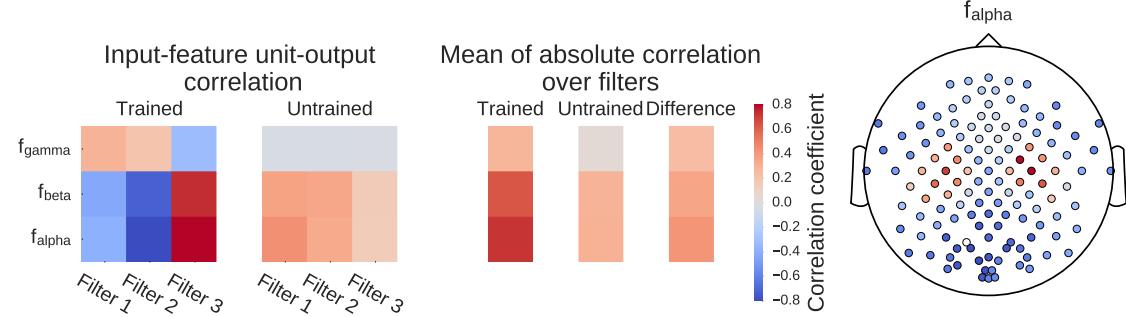
To compute a single scalp plot for a frequency band, we computed the mean of the absolute correlations over all units for each convolutional filter and each electrode for that frequency band. To disentangle effects which are caused by the training of the network from those caused by the architecture, we computed the scalp plot for a trained and an untrained model. The scalp plot for a subject is then the scalp plot of the trained model minus the scalp plot of the untrained model (see Figure S1b). The group scalp plot is the mean of these differential scalp plots over all subjects.

To compare the resulting maps against scalp maps that simply result from class-feature correlations, we also computed the linear correlation between mean squared envelope values and the one-hot-encoded classes, in the same way as before. First, for each trial, each sensor and each frequency band, we constructed a vector of the moving window squared envelope values as before (with the moving window now the size of the receptive field of the last layer of the ConvNet). Second, for each trial and each class, we constructed a vector of either value 1 if the trial was of the given class or value 0 if it was of another class. The concatenated vectors then resulted in a time series with value 1 if the time point belonged to a given class and value 0 if it did not. Then we correlated the moving window squared envelope time series vectors with the class time series vectors, resulting in one correlation value per class, sensor and frequency band combination. As in the other computations, we subtracted the correlations resulting from predictions of an untrained deep ConvNet.

A further question is whether the correlations could be a result of the unit outputs encoding the final class label. Such correlations could also result from using other discriminative features than the features we analyzed. To investigate this question, we correlated the unit outputs for each layer

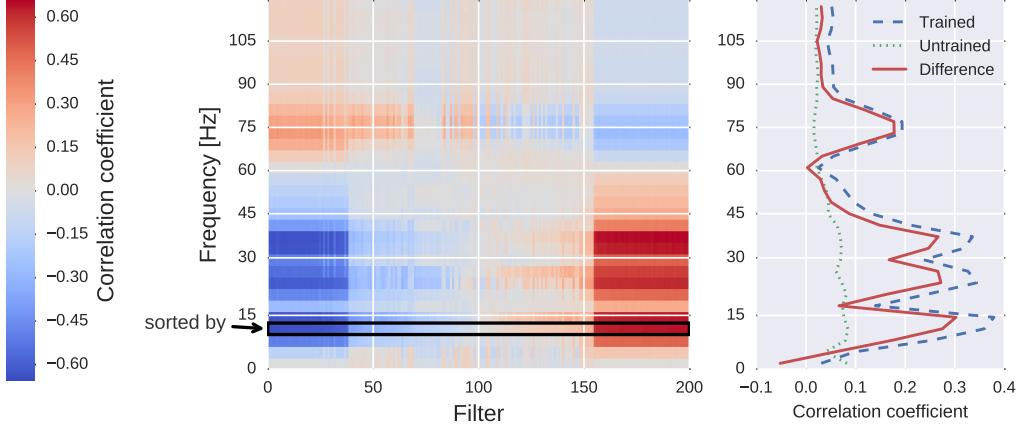


(a) Feature inputs and unit outputs for input-feature unit-output correlation map. Moving average of squared envelopes and unit outputs for 10 trials. Upper rows show mean squared envelopes over the receptive field for three frequency ranges in the alpha, beta and gamma frequency band, standardized per frequency range. Lower rows show corresponding unit outputs for three filters, standardized per filter. All time series standardized for the visualization.



(b) Input-feature unit-output correlations and corresponding scalp map for the alpha band. Left: Correlation coefficients between unit outputs of three filters and mean squared envelope values over the corresponding receptive field of the units for three frequency ranges in the alpha (7–13 Hz), beta (13–31 Hz), and gamma (71–91 Hz) frequency band. Results are shown for the trained and the untrained ConvNet and for one electrode. Middle: Mean of the absolute correlation coefficients over the three filters for the trained and the untrained ConvNet, and the difference between trained and untrained ConvNet. Right: An exemplary scalp map for correlations in the alpha band (7–13 Hz), where the color of each dot encodes the correlation difference between a trained and an untrained ConvNet for one electrode. Note localized positive effects above areas corresponding to the right and left sensorimotor hand/arm areas, indicating that activity in these areas has large absolute correlations with the predictions of the trained ConvNet.

Figure S1: Computation overview for input-feature unit-output network correlation map.

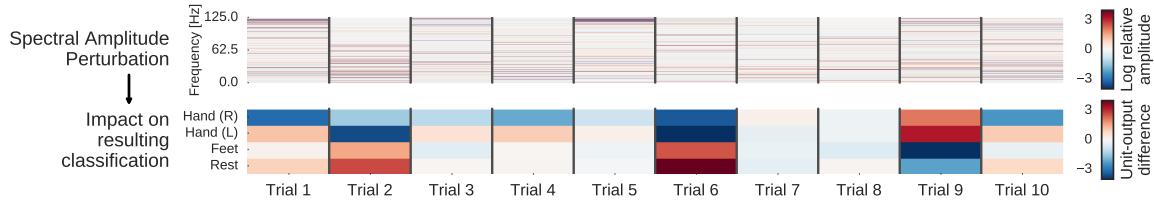


**Figure S2: Correlation between the mean squared envelope feature and unit output for a single subject at one electrode position (FCC4h).** Left: All correlations. Colors indicate the correlation between unit outputs per convolutional filter (x-axis) and mean squared envelope in different frequency bands (y-axis). Filters are sorted by their correlation to the 7–13 Hz envelope (outlined by the black rectangle). Note the large correlations/anti-correlations in the alpha/beta bands (7–31 Hz) and somewhat weaker correlations/anti-correlations in the gamma band (around 75 Hz). Right: Mean absolute values across units of all convolutional filters for all correlation coefficients of the trained model, the untrained model and the difference between the trained and untrained model. Peaks in the alpha, beta and gamma bands are clearly visible.

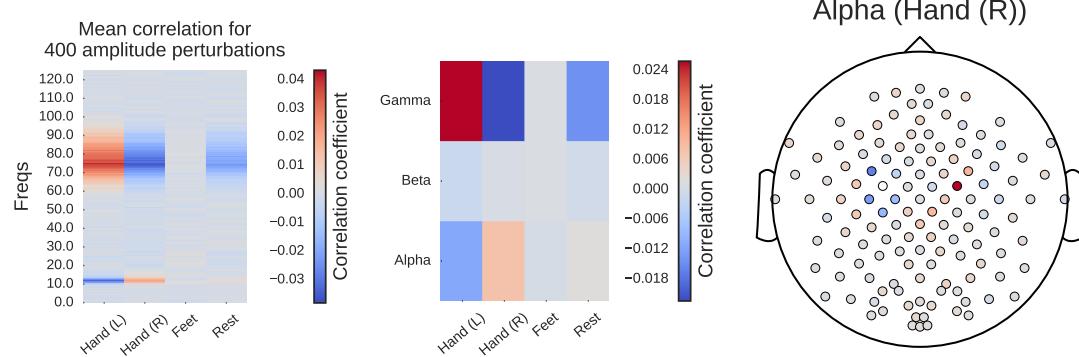
with the class labels. Here, we proceeded the same way as described in the previous paragraph, but correlated unit outputs directly with class labels. We then computed a single absolute correlation coefficient per layer in two ways: First, we computed the mean absolute correlation coefficient for all classes and all filters. These correlations should show how strongly the unit outputs encode the class labels on average across filters. Second, we computed the maximum absolute correlation coefficients for each class over all filters and then computed the mean of these maxima of the four classes. These correlations should show how strongly the unit outputs encode the class labels for the most “class-informative” filters. Finally, for both versions and as in the other visualizations, we also computed the difference of these correlations between a trained and an untrained model. In summary, this approach allowed to show how unit-output class correlations arise from layer to layer through the ConvNet.

#### A.5.2 Input-perturbation network-prediction correlation map

To investigate the causal effect of changes in power on the deep ConvNet, we correlated changes in ConvNet predictions with changes in amplitudes by perturbing the original trial amplitudes (see figure S3 for an overview). Concretely, we transformed all training trials into the frequency domain by a Fourier transformation. Then we randomly perturbed the amplitudes by adding Gaussian noise (with mean 0 and variance 1) to them. The phases were kept unperturbed. After the perturbation, we re-transformed to the time domain by the inverse Fourier transformation. We computed predictions of the deep ConvNet for these trials before and after the perturbation (predictions here refers to outputs of the ConvNet directly before the softmax activation). We repeated this procedure with 400



(a) Example spectral amplitude perturbation and resulting classification difference. Top: Spectral amplitude perturbation as used to perturb the trials. Bottom: unit-output difference between unperturbed and perturbed trials for the classification layer units before the softmax.



(b) Input-perturbation network-prediction correlations and corresponding network correlation scalp map for alpha band. Left: Correlation coefficients between spectral amplitude perturbations for all frequency bins and differences of the unit outputs for the four classes (differences between unperturbed and perturbed trials) for one electrode. Middle: Mean of the correlation coefficients over the the alpha (7–13 Hz), beta (13–31 Hz) and gamma (71–91 Hz) frequency ranges. Right: An exemplary scalp map for the alpha band, where the color of each dot encodes the correlation of amplitude changes at that electrode and the corresponding prediction changes of the ConvNet. Negative correlations on the left sensorimotor hand/arm areas show an amplitude decrease in these areas leads to a prediction increase for the Hand (R) class, whereas positive correlations on the right sensorimotor hand/arm areas show an amplitude decrease leads to a prediction decrease for the Hand (R) class. This complements the information from the input-feature unit-output network correlation map (see Figure S1b), which showed band power in these areas is strongly correlated with unit outputs in the penultimate layer.

Figure S3: Computation overview for input-perturbation network-prediction correlation map.

perturbations sampled from aforementioned Gaussian distribution and then correlated the change in input amplitudes (i.e., the perturbation/noise we added) with the change in the ConvNet predictions. To ensure that the effects of our perturbations reflect the behavior of the ConvNet on realistic data, we also checked that the perturbed input does not cause the ConvNet to misclassify the trials (as can easily happen even from small perturbations, see [Szegedy et al. \(2014\)](#)). For that, we computed accuracies on the perturbed trials. For all perturbations of the training sets of all subjects, accuracies stayed above 99.5% of the accuracies achieved with the unperturbed data.

### A.5.3 EEG spectral power topographies

To visualize the class-specific EEG spectral power modulations, we computed band-specific envelope-class correlations in the alpha, beta and gamma bands for all classes of the High-Gamma Dataset. The group-averaged topographies of these maps could be readily compared to our input-feature unit-output network correlation maps, since, similar to the power-class correlation map described in Section A.5.1, we computed correlations of the moving average of the squared envelope with the actual class labels, using the receptive field size of the final layer as the moving average window size. Since this is a ConvNet-independent visualization, we did not subtract any values of an untrained ConvNet. We show the resulting scalp maps for the four classes and did not average over them. Note that these computations were only used for the power topographies shown in Figure 11 and did not enter the decoding analyses as described in the preceding sections.

## A.6 Dataset details

The BCI Competition IV dataset 2a (from here on referred to as BCI Competition Dataset) is a 22-electrode EEG motor-imagery dataset, with 9 subjects and 2 sessions, each with 288 four-second trials of imagined movements per subject (movements of the left hand, the right hand, the feet and the tongue) ([Brunner et al., 2008](#)). The training set consists of the 288 trials of the first session, the test set of the 288 trials of the second session.

Our “High-Gamma Dataset” is a 128-electrode dataset (of which we later only use 44 sensors covering the motor cortex, (see Section 2.7.1), obtained from 20 healthy subjects (9 female, 4 left-handed, age  $27.5 \pm 3.2$  (mean $\pm$ std)) with roughly 1000 ( $992 \pm 135.3$ , mean $\pm$ std) four-second trials of executed movements divided into 13 runs per subject. The four classes of movements were movements of either the left hand, the right hand, both feet, and rest (no movement, but same type of visual cue as for the other classes). The training set consists of the approx. 880 trials of all runs except the last two runs, the test set of the approx. 160 trials of the last 2 runs. This dataset was acquired in an EEG lab optimized for non-invasive detection of high-frequency movement-related EEG components ([Ball et al., 2008; Darvas et al., 2010](#)). Such high-frequency components in the range of approx. 60 to above 100 Hz are typically increased during movement execution and may contain useful movement-related information ([Crone et al., 1998; Hammer et al., 2016; Quandt et al., 2012](#)). Our technical EEG Setup comprised (1.) Active electromagnetic shielding: optimized for frequencies from DC - 10 kHz (-30 dB to -50 dB), shielded window, ventilation & cable feedthrough (mrShield, CFW EMV-Consulting AG, Reute, CH) (2.) Suitable amplifiers: high-resolution (24 bits/sample) and low-noise ( $<0.6 \mu\text{V}$  RMS 0.16–200 Hz,  $<1.5 \mu\text{V}$  RMS 0.16–3500 Hz), 5 kHz sampling rate (NeurOne, Mega Electronics Ltd, Kuopio, FI) (3.) actively shielded EEG caps: 128 channels (WaveGuard Original, ANT, Enschede, NL) and (4.) full optical decoupling: All devices are battery powered and communicate via optic fibers.

Subjects sat in a comfortable armchair in the dimly lit Faraday cabin. The contact impedance from electrodes to skin was typically reduced below 5 kOhm using electrolyte gel (SUPER-VISC, EASYCAP GmbH, Herrsching, GER) and blunt cannulas. Visual cues were presented using a monitor outside the cabin, visible through the shielded window. The distance between the display and the subjects' eyes was approx. 1 m. A fixation point was attached at the center of the screen. The subjects were instructed to relax, fixate the fixation mark and to keep as still as possible during the motor execution task. Blinking and swallowing was restricted to the inter-trial intervals.

The tasks were as following. Depending on the direction of a gray arrow that was shown on black background, the subjects had to repetitively clench their toes (downward arrow), perform sequential finger-tapping of their left (leftward arrow) or right (rightward arrow) hand, or relax (upward arrow). The movements were selected to require little proximal muscular activity while still being complex enough to keep subjects involved. Within the 4-s trials, the subjects performed the repetitive movements at their own pace, which had to be maintained as long as the arrow was showing. Per run, 80 arrows were displayed for 4 s each, with 3 to 4 s of continuous random inter-trial interval. The order of presentation was pseudo-randomized, with all four arrows being shown every four trials. Ideally 13 runs were performed to collect 260 trials of each movement and rest. The stimuli were presented and the data recorded with BCI2000 ([Schalk et al., 2004](#)). The experiment was approved by the ethical committee of the University of Freiburg.

### A.7 EEG preprocessing

We resampled the High-Gamma Dataset to 250 Hz, i.e., the same as the BCI Competition Dataset, to be able to use the same ConvNet hyperparameter settings for both datasets. To ensure that the ConvNets only have access to the same frequency range as the CSPs, we low-pass filtered the BCI Competition Dataset to below 38 Hz. In case of the 4- $f_{end}$ -Hz dataset, we highpass-filtered the signal as described in [2.7.1](#) (for the BCI Competition Dataset, we *bandpass-filtered* to 4-38 Hz, so the previous lowpass-filter step was merged with the highpass-filter step). Afterwards, for both sets, for the ConvNets, we performed electrode-wise exponential moving standardization with a decay factor of 0.999 to compute exponential moving means and variances for each channel and used these to standardize the continuous data. Formally,

$$x't = (x_t - \mu_t)/\sqrt{\sigma_t^2} \quad (4)$$

$$\mu_t = 0.001x_t + 0.999\mu_{t-1} \quad (5)$$

$$\sigma_t^2 = 0.001(x_t - \mu_t)^2 + 0.999\sigma_{t-1}^2 \quad (6)$$

where  $x't$  and  $x_t$  are the standardized and the original signal for one electrode at time  $t$ , respectively. As starting values for these recursive formulas we set the first 1000 mean values  $\mu_t$  and first 1000 variance values  $\sigma_t^2$  to the mean and the variance of the first 1000 samples, which were always completely inside the training set (so we never used future test data in our preprocessing). Some form of standardization is a commonly used procedure for ConvNets; exponentially moving standardization has the advantage that it is also applicable for an online BCI. For FBCSP, this standardization always worsened accuracies in preliminary experiments, so we did not use it. We also did not use the standardization for our visualizations to ensure that the standardization does not make our visualizations harder to interpret. Overall, the minimal preprocessing without any manual feature extraction ensured our end-to-end pipeline could in principle be applied to a large number of brain-signal decoding tasks.

We also only minimally cleaned the data sets to remove extreme high-amplitude recording artifacts. Our cleaning method thus only removed trials where at least one channel had a value outside  $\pm 800 \mu\text{V}$ . We kept trials with lower-amplitude artifacts as we assumed these trials might still contain useful brain-signal information. As described below, we used visualization of the features learned by the ConvNets to verify that they learned to classify brain signals and not artifacts. Furthermore, for the High-Gamma Dataset, we used only those sensors covering the motor cortex: all central electrodes (45), except the Cz electrode which served as the recording reference electrode. Interestingly, using all electrodes led to worse accuracies for both the ConvNets and FBCSP, which may be a useful insight for the design of future movement-related decoding/BCI studies. Any further data restriction (trial- or channel-based cleaning) never led to accuracy increases in either of the two methods when averaged across all subjects. For the visualizations, we used all electrodes and common average re-referencing to investigate spatial distributions for the entire scalp.

## A.8 Statistics

We used Wilcoxon signed-rank tests to check for statistical significance of the mean difference of accuracies between decoding methods ([Wilcoxon, 1945](#)). We handled ties by using the average rank of all tied data points and zero differences by assigning half of the rank-sum of these zero differences to the positive rank-sums and the other half to the negative rank-sums. In case of a non-integer test-statistic value caused by ties or zeros, we rounded our test-statistic to the next larger integer, resulting in a more conservative estimate.

## A.9 Software implementation and hardware

We performed the ConvNet experiments on Geforce GTX Titan Black GPUs with 6 GB memory. The machines had Intel(R) Xeon(R) E5-2650 v2 CPUs @ 2.60 GHz with 32 cores (which were never fully used as most computations were performed on the GPU) and 128 GB RAM. FBCSP was computed on an Intel(R) Xeon(R) CPU E5-2650 v2 @ 2.60 GHz with 16 cores and 64 GB RAM. We implemented our ConvNets using the Lasagne framework ([Dieleman et al., 2015](#)), preprocessing of the data and FBCSP were implemented with the Wyrm library ([Ventur et al., 2015](#)). We will make the code to reproduce these results publicly available.