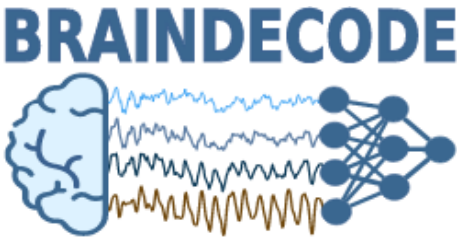
# DEEP LEARNING FOR BRAIN-SIGNAL DECODING FROM ELECTROENCEPHALOGRAPHY

ROBIN TIBOR SCHIRRMEISTER




June 2023

[ June 2, 2023 at 23:56 – classicthesis v4.6 ]

# CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LISTINGS

# ACRONYMS

# 1 INTRODUCTION

**Deep Learning (DL) is a very promising method to decode brain signals from EEG**

- Deep Learning (DL) is a very promising method to decode brain signals from EEG
- Deep learning may extract useful information from EEG signals that humans cannot
- Deep learning may improve EEG-based diagnosis, enable new assistive technologies and advance scientific understanding of the EEG signal
- Our EEG decoding deep learning models perform as good or better than feature-based methods on a wide range of tasks
- Visualizations using convolutional and invertible networks reveal neurophysiologically plausible as well as surprising learned EEG features

Machine learning (ML), i.e., using data to learn programs that perform a desired task, has the potential to benefit medical brain-signal-decoding applications. Compared to humans, machine-learning programs can process larger amounts of brain-signal data and may extract different information. For example, machine-learning algorithms have been developed to help doctors triage patients by quickly detecting stroke biomarkers from computed tomography (CT) [15], to enable brain-computer interfaces by recognizing people's intentions from electroencephalographic (EEG) in real time [1] and to detect pathology from long brain signal recordings [25, 75]. Also, as brain signals are far from being fully understood, machine-learning algorithms have the potential to advance scientific understanding by finding new brain-signal biomarkers for different pathologies [64].

Electroencephalographic (EEG) brain-signal recordings are well-suited for machine learning since they are easy to acquire, while being time-consuming and challenging to manually interpret by doctors. Generating large EEG datasets is relatively simple compared to other medical recordings because of the low cost and minimal side effects of performing EEG recordings. Furthermore, EEG recordings are particularly challenging for humans to interpret, making them a promising target for information extraction through machine learning. Some clinical applications of EEG such as pathology diagnosis may be improved by machine learning, while others such as brain-computer interfaces are even only possible because of it. Finally, since the

information contained in the EEG signal is far from being fully understood, machine learning may even help understand the EEG signal itself better.

Deep learning is a very promising approach for brain-signal decoding from EEG. The term deep learning describes machine-learning models with multiple computational stages, where the computational stages are typically trained jointly to solve a given task [43, 78]. Convolutional neural networks (ConvNets) are deep learning models that are inspired by computational structures in the visual cortex of the brain. ConvNets only have very general assumptions about the properties of their training signals embedded into them (such as smoothness and local-to-global hierarchical structure) and have shown great success on a variety of decoding tasks on natural signals, including object detection in images, speech recognition from audio or machine translation. Therefore, ConvNets are very promising to apply to hard-to-understand natural signals like EEG signals.

Prior to the work presented in this thesis, it was unclear how well ConvNet architectures can decode EEG signals compared to hand-engineered, feature-based approaches. The high dimensionality, low signal-to-noise ratio and large signal variability (e.g., from person to person or even session to session for the same person) of EEG data present challenges that may be better addressed by feature-based approaches that exploit more specific assumptions about the EEG signal. While there had been previous efforts to apply deep learning to EEG, a systematic study of the performance of modern ConvNets on EEG decoding compared with a strong feature-based baseline and including the impact of network architecture and training hyperparameters, was lacking. Furthermore, research into understanding what features the ConvNets extract from the EEG signal had been limited.

We therefore created several ConvNet architectures to thoroughly evaluate on EEG decoding. We first evaluated our ConvNet's decoding performance under a range of different hyperparameter choices on widely studied movement-related decoding tasks like decoding which limb a person is thinking of moving. On those tasks, we found our ConvNets to perform at least as good as a strong feature-based baseline. The ConvNets also generalized well to a range of other decoding tasks, including other mental imageries, decoding whether a person made or perceived an error, as well as pathology diagnosis.

We also developed visualizations to understand the features the ConvNets extract from the EEG signal, finding that their predictions are sensitive to plausible neurophysiological features. Using perturbations of spectral features like amplitude and phase, we show topographies of the causal effects of spectral changes on the networks predictions. For decoding of executed movements, these topographies are consistent with known movement-related spectral brain-signal changes like contralateral alpha power decreases (e.g., decrease in alpha power on the right side of the head when moving the left hand). They also suggest that networks learn to use high-gamma information to predict the performed movement. Visualizations of inputs that maxi-

mally activate specific units in one of our ConvNets further reveals that the ConvNet has also learned specific timecourses of amplitude changes, going beyond using just averaged spectral features.

Later, we more deeply investigated features learned for pathology decoding. Here, we made use of invertible networks, networks that are designed such that you can invert intermediate and final network outputs back to a corresponding input, allowing to visualize output changes in input space. Further, we also developed a smaller network that is specifically designed to be interpretable and train it to mimic the invertible network. Using these methods we could directly show and investigate temporal waveforms with spatial topographies that are associated with pathological or healthy recordings. These visualizatiosn revealed both neurophysiologically plausible features like temporal slowing as a marker for a pathological EEG or occipital alpha as a marker for healthy EEG, as well as surprising features like frontal and temporal very-low-frequency 0.5 Hz components.

In this thesis, I will first describe the research on deep learning EEG decoding prior to our work, then proceed to describe the deep network architectures and training methods we developed to rival or surpass feature-based EEG decoding approaches on movement- and other task-related EEG decoding tasks as well as pathology diagnosis. Furthermore, I also describe the visualization methods we developed that suggest the networks are using plausible neurophysiological patterns to solve their tasks. In two separate method and result chapters, I will delve more deeply into understanding the learned features for pathology decoding, including by using invertible networks. Finally, I conclude with my thoughts on the current state of EEG deep learning decoding and promising avenues for further work like cross-dataset decoding models as well as models that can process larger timescales of EEG signals.

TODO: textbox

# 2 PRIOR WORK

Prior to our work research on deep-learning-based EEG decoding was limited

- Few studies compared to published feature-based decoding results
- Most EEG DL architectures had only 1-3 convolutional layers and included fully-connected layers with many parameters
- Most work only considered very restricted frequency ranges
- Most studies only compared few design choices and training strategies

Prior to 2017, when the first work presented in this thesis was published, there was only limited literature on EEG decoding with deep learning. In this chapter, I outline what decoding problems, input representations, network architectures, hyperparameter choices and visualizations were evaluated in prior work. This is based on the literature research that we presented in Schirrmeister et al. [77].

## 2.1 DECODING PROBLEMS AND BASELINES

| DECODING PROBLEM | NUMBER OF STUDIES | PUBLISHED BASELINE |
|---|---|---|
| Imagined or Executed Movement | 6 | 2 |
| Oddball/P300 | 5 | 1 |
| Epilepsy-related | 4 | 2 |
| Music Rhythm | 2 | 0 |
| Memory Performance/Cognitive Load | 2 | 0 |
| Driver Performance | 1 | 0 |

**Table 2.1: Decoding problems in deep-learning EEG decoding studies prior to our work.** Studies with published baseline compared their decoding results to an external baseline result published by other authors.

The most widely studied decoding problems were movement-related decoding problems such as decoding which body part (hand, feet etc.) a person is moving

or imagining to move (see Table 2.1). From the 19 studies we identified at the time, only 5 compared their decoding results to an external published baseline result, limiting the insights about deep-learning EEG decoding performance. We therefore decided to compare deep-learning EEG decoding to a strong feature-based baseline (see Chapter 3) on widely researched movement-related decoding tasks.

## 2.2 INPUT DOMAINS AND FREQUENCY RANGES

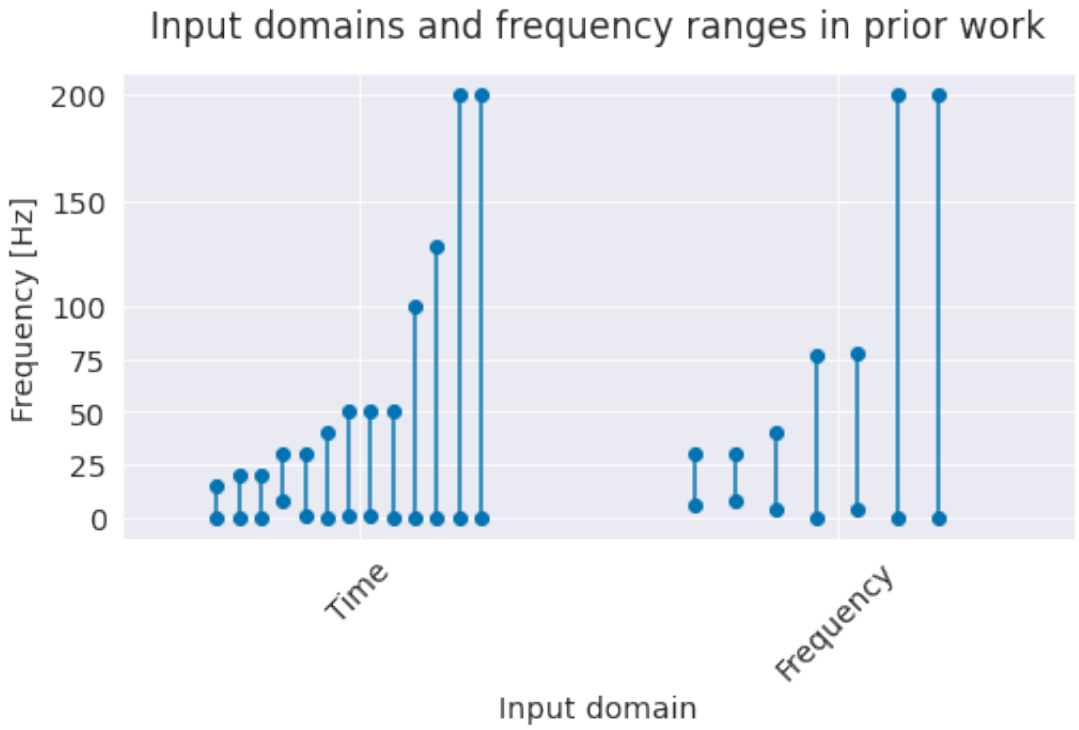



**Figure 2.1: Input domains and frequency ranges in prior work.** Grey linesrepresent frequency ranges of individual studies. Note that many studies only include frequencies below 50 Hz, some use very restricted ranges (alpha/beta band)

Deep networks can either decode directly from the time-domain EEG or process the data in the frequency domain, for example after a Fourier transformation. 12 of the prior studies used time-domain inputs, 6 used frequency-domain inputs and one used both. We decided to work directly in the time domain, as the deep networks should in principle be able to learn how to extract any needed spectral information from the time-domain input.

Most prior studies that were working in the time domain only used frequencies below 50 Hz. We were interested in how well deep networks can also extract higher-

frequency components of the EEG signal. For that, we used a sampling rate of 250 Hz, which means we were able to analyze frequencies up to the Nyquist frequency of 125 Hz. As a suitable dataset for high-frequency analysis, we included our high-gamma dataset in our study, since it was recorded specifically to allow extraction of higher-frequency (>50 Hz) information from scalp EEG [77].

## 2.3 NETWORK ARCHITECTURES

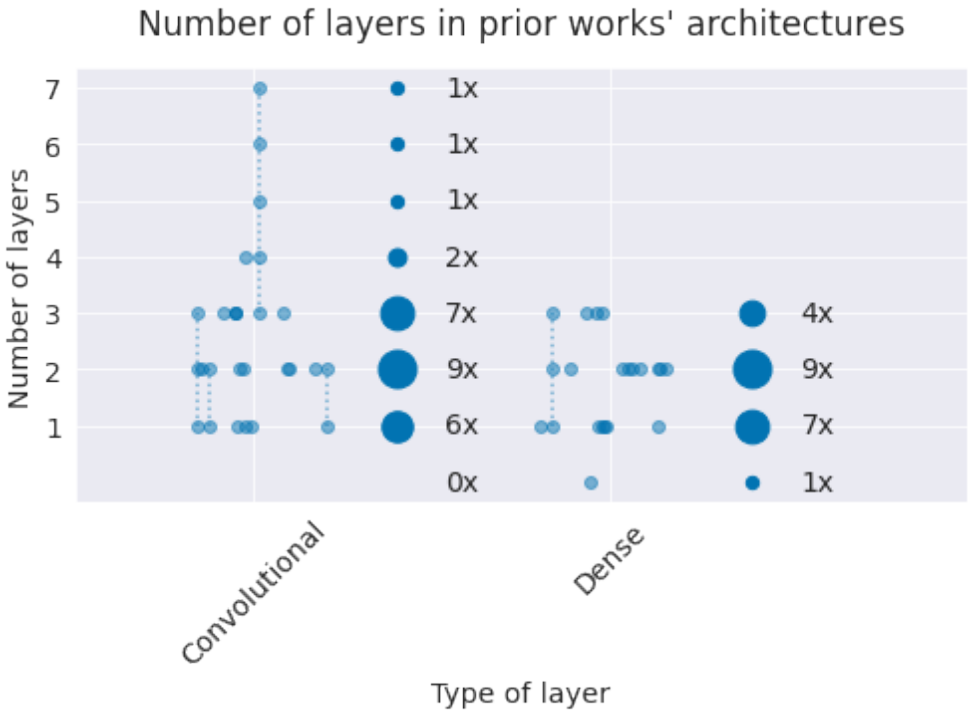



**Figure 2.2: Number of layers in prior work.** Small grey markers represent individual architectures. Dashed lines indicate different number of layers investigated in a single study (e.g., a single study investigated 3-7 convolutional layers). Larger grey markers indicate sum of occurences of that layer number over all studies (e.g., 9 architectures used 2 convolutional layers). Note most architectures use only 1-3 convolutional layers.

The architectures used in prior work typically only included up to 3 layers, with only 2 studies considering more layers. As network architectures in other domains tend to be a lot deeper, we also evaluated architectures with a larger number of layers in our work. Several architectures from prior work also included fully-connected layers with larger number of parameters which had fallen out of favor in computer-vision deep-

learning architectures due to their large compute and memory requirements with little accuracy benefit. Our architectures do not include traditional fully-connected layers with a large number of parameters.

## 2.4 HYPERPARAMETER EVALUATIONS

Prior work varied widely in their comparison of design choices and training strategies. 6 of the studies did not compare any design choices or training strategy hyperparameters. The other 13 studies evaluated different hyperparameters, with the most common one the kernel size (see Table 2.2). Only one study evaluated a wider range of hyperparameters [85]. To fill this gap, we compared a wider range of design choices and training strategies and specifically evaluated whether improvements of computer vision architecture design choices and training strategies also lead to improvements in EEG decoding.

## 2.5 VISUALIZATIONS

Visualizations can help understand what information the networks are extracting from the EEG signal. 11 of the prior 19 studies presented any visualizations. These studies mostly focused on analyzing weights and activations, see Table 2.3. In our work, we first focused on investigating how far the networks extract spectral features known to work well for movement-related decoding, see Chapter 6. Later, we also developed more sophisticated visualization methods and applied them both to pathology decoding, see Chapter 7 and Chapter 11.

Open Questions

- How do ConvNets perform on well-researched EEG movement-related decoding tasks against strong feature-based baselines?
- How do shallower and deeper architectures compare?
- How do design choices and training strategies affect the decoding performance?
- What features do the deep networks learn on the EEG signals?
- Do they learn to use higher-frequency (>50 Hz) information?

| STUDY | DESIGN CHOICES | TRAINING STRATEGIES |
|---|---|---|
| [42] | Kernel sizes | |
| [87] | | Different time windows |
| [90] | Addition of six-layer stacked autoencoder on ConvNet features Kernel sizes | |
| [46] | | Different subdivisions of frequency range Different lengths of time crops Transfer learning with auxiliary non-epilepsy datasets |
| [28] | Replacement of convolutional layers by restricted Boltzmann machines with slightly varied network architecture} | |
| [3] | 1 or 2 convolutional layers | |
| [59] | | Cross-subject supervised training, within-subject finetuning of fully connected layers |
| [6] | Number of convolutional layers Temporal processing of ConvNet output by max pooling, temporal convolution, LSTM or temporal convolution + LSTM | |
| [84] | Kernel sizes | Pretraining first layer as convolutional autoencoder with different constraints |
| [73] | Combination ConvNet and MLP (trained on different features) vs. only ConvNet vs. only MLP | |
| [85] | 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 |
| [99] | Partially supervised CSA | |
| [14] | Electrode subset (fixed or automatically determined) Using only one spatial filter Different ensembling strategies | |

**Table 2.2: Design choices and training strategies of prior work.**

| STUDY | TYPE(S) | FINDINGS |
|---|---|---|
| [87] | Weights (spatial) | Largest weights found over prefrontal and temporal cortex. |
| [51] | Weights, activations, gradient-based saliency maps | Weights showed typical P300 distribution. Activations were high at plausible times (300-500ms). Saliency maps showed plausible spatio-temporal plots. |
| [90] | Weights (spatial + frequential) | Some weights represented difference of values of two electrodes on different sides of head. |
| [46] | Weights, clustering of weights | Clusters of weights showed typical frequency band subdivision (delta, theta, alpha, beta, gamma). |
| [3] | Weights, correlation weights and interictal epileptic discharges (IED), activations | Weights increasingly correlated with IED waveforms with increasing number of training iterations. Second layer captured more complex and well-defined epileptic shapes than first layer. IEDs led to highly synchronized activations for neighbouring electrodes. |
| [93] | Input occlusion and effect on prediction accuracy | Allowed to locate areas critical for seizure. |
| [80] | Weights (spatial) | Some filter weights had expected topographic distributions for P300, others filters had large weights on areas not traditionally associated with P300. |
| [6] | Inputs that maximally activate given filter; Activations of these inputs, "Deconvolution" for these inputs | Different filters were sensitive to different frequency bands. Later layers had more spatially localized activations. Learned features had noticeable links to well-known electrophysiological markers of cognitive load. |
| [84] | 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. |
| [50] | Weights; Mean and single-trial activations | Spatiotemporal regularization led to softer peaks in weights. Spatial weights showed typical P300 distribution; Activations mostly had peaks at typical times (300-400ms). |
| [14] | Weights | Spatial filters were similar for different architectures. Spatial filters were different (more focal, more diffuse) for different subjects. |

**Table 2.3: Visualizations presented in prior work.**

# 3 FILTER BANK COMMON SPATIAL PATTERNS AND FILTERBANK NETWORK

Filter Bank Common Spatial Patterns (FBCSP) is a strong feature-based baseline

- Learns spatial topography of task-related spectral power changes
- Widely used for (movement-related) EEG-decoding
- We used it as a starting point for our network architecture development

In a prior master thesis [76], we had developed a neural network architecture closely resembling the feature-based decoding algorithm filter bank common spatial patterns. In this chapter, I describe filter bank common spatial patterns as well as the corresponding filter bank network of the prior master thesis as the starting point for the network architectures developed in the context of this thesis.

## 3.1 FILTER BANK COMMON SPATIAL PATTERNS AS A STARTING POINT

We selected filter bank common spatial patterns (FBSCP [2, 16]) as the feature-based EEG-decoding algorithm we were trying to imitate in our initial neural network architectures. FBCSP is an EEG-decoding algorithm that has been successfully used in task-related EEG-decoding competitions [91]. FBCSP aims to decode task-related changes in signal amplitude in different frequencies, such as a decrease in the amplitude of alpha and beta-band oscillations during movement imagination. In the following, we will explain how FBCSP decodes two classes of EEG signals by finding frequency-specific spatial filters that transform the signal, such that the transformed signal has relatively high variance for one class and low variance for the other class and vice versa.

## 3.2 COMMON SPATIAL PATTERNS

The basic building block of FBCSP is the common spatial patterns (CSP) algorithm. CSP is used to decode neuronal activity that leads to a change in the amplitudes of the EEG signal with a specific spatial topography [11, 41, 65]. To do that, CSP

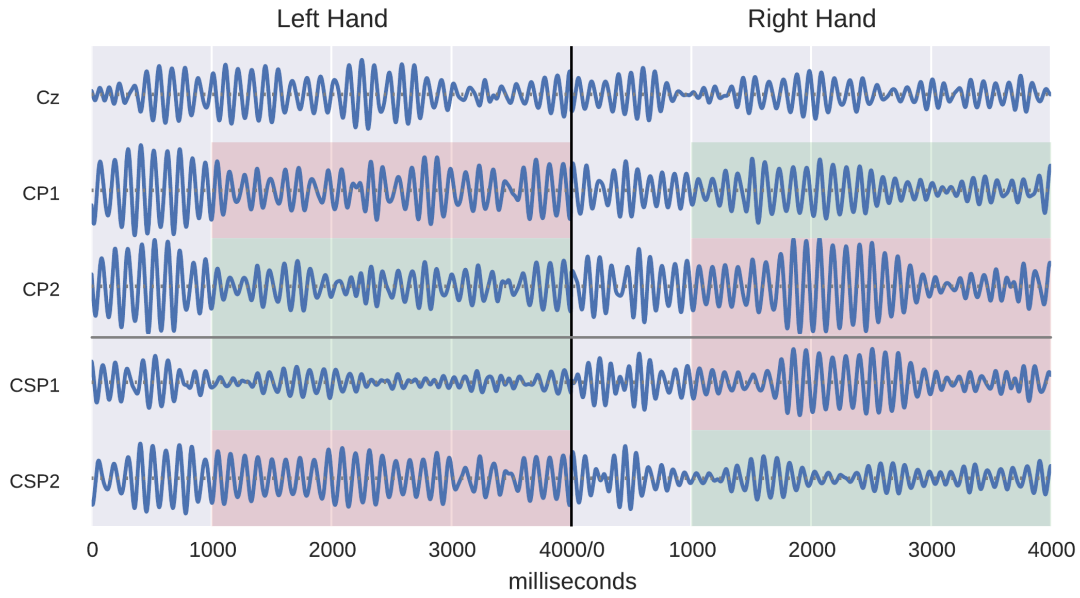



**Figure 3.1: Common Spatial Patterns example.** Top parts show EEG signals for three electrodes during a left hand and a right hand movement. Bottom parts show spatially filtered signals of two CSP filters. Green parts have lower variance and red parts have higher variance. Note that this difference is strongly amplified after CSP filtering. Figure from prior master thesis [76].

aims to maximize the ratio of the signal variance between spatially filtered signals of two classes, e.g. of the signal during two different movements. For example, the signal of a spatial filter computed by CSP may have a very large variance during movements of the left hand and a very small variance during movements of the right hand. Concretely, we are given signals $X_1, X_2 \in \mathbb{R}^{nxkxt}$ from $n$ EEG trials (can be different for $X_1, X_2$, $k$ EEG electrodes and $t$ timepoints within each trial. CSP then finds a spatial filter $w$ that maximize the ratio of the variances of the spatially filtered $X_1, X_2$:

$$w = \arg\max_w \frac{Var(w^T X_1)}{Var(w^T X_2)} = \arg\max_w \frac{||w^T X_1||^2}{||w^T X_2||^2} = \arg\max_w \frac{w^T X_1 X_1^T w}{w^T X_2 X_2^T w}$$

Rather than just finding a single spatial filter $w$, CSP is typically used to find a whole matrix of spatial filters $W^{kxk}$, with spatial filters ordered by the above variance ratio and orthogonal to each other. The first filter $w_1$ results in the largest variance ratio and the last filter $w_k$ results in the smallest variance ratio. Different algorithms can then be used to subselect some set of filters to filter signals for a subsequent decoding algorithm.

The CSP-filtered signals can be used to construct features to train a classifier. Since the CSP-filtered signals should have very different variances for the different classes, the natural choice is to use the per-trial variances of the CSP-filtered signals as features. This results in as many features per trial as the number of CSP filters that were selected for decoding. Typically, one applies the logarithm to the variances to get more standard-normally distributed features.

## 3.3 FILTER BANK COMMON SPATIAL PATTERNS

CSP is typically applied to an EEG signal that has been bandpass filtered to a specific frequency range. The filtering to a frequency range is useful as brain signals cause EEG signal amplitude changes that are temporally and spatially different for different frequencies [2]. For example, during movement the alpha rhythm may be suppressed for multiple electrodes covering a fairly large region on the scalp while the high gamma rhythm would be amplified for a few electrodes covering a smaller region.

Filter bank common spatial patterns applies CSP separately on signals bandpass-filtered to different frequency ranges [2, 16]. This allows to capture multiple frequency-specific changes in the EEG signal and can also make the decoding more robust to subject-specific signal characteristics, i.e., which frequency range is most informative for a given subject. The trial-log-variance features of each frequencyband and each CSP filter are then concatenated to form the entire trial feature vector. Typically, a feature selection procedure will select a subset of these features to train the final classifier.

The overall FBCSP pipeline hence looks like this:

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 labeled trials, e.g., 4-second left-hand or right-hand movement windows.
3. **CSP computation**: Per frequency band, the common spatial patterns (CSP) algorithm is applied to extract spatial filters (see Section 3.2).
4. **Spatial filtering**: The spatial filters computed in Step 2 are applied to the EEG signal.
5. **Feature construction**: Feature vectors 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. **Feature selection**: A feature selection algorithm may be used to only retain a subset of the features for classification.

7. **Classification**: A classifier is trained to predict per-trial labels based on the feature vectors.

## 3.4 FILTER BANK NETWORK ARCHITECTURE

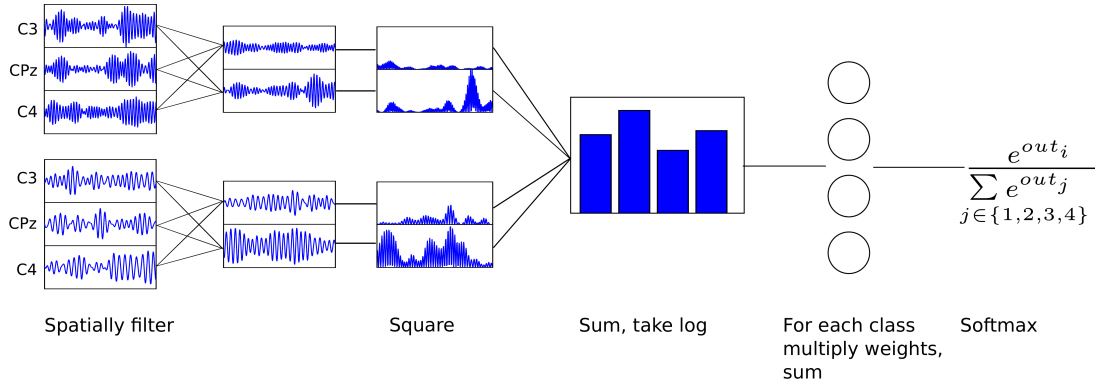



**Figure 3.2: Filter bank network architecture overview.** Input signals were bandpass filtered to different frequency ranges. Signals are first transformed by learned spatial filters, then squared, summed and the log-transformed. The resulting features are transformed into class probabilities by a classification weights followed by the softmax function. Figure taken from a master thesis [76].

The first neural network architecture was developed by us in a prior master thesis [76] to jointly learn the same steps that are learned separately by FBCSP (see Figure 3.2). Concretely, the network simultaneously learn the spatial filters across many frequency bands and the classification weights for the log squared sums of all resulting spatially filtered signals. To be able to do that, the network is fed with input EEG signals that were bandpass-filtered to different frequency ranges. The network then performs the following steps:

**Spatial Filtering**

$$h_1 = W_s^T x \qquad \text{Apply spatial filter weights } W_s \text{ to inputs}$$

**Feature Construction**

$$h_2 = h_1^2 \qquad \text{Square the spatially filtered signals}$$
$$h_3 = \sum_t (h_2) \qquad \text{Sum the squared signals across trial timepoints } t$$
$$h_4 = \log(h_3) \qquad \text{Take the logarithm of the summed values}$$

**Classification**

$$h_5 = W_c^T h_4 \qquad \text{Apply classifier weights } W_c \text{ on these features}$$

$$p(c_i|h_5) = \frac{e^{h_{5,i}}}{\sum_j e^h_{5,j}} \qquad \text{Take the softmax to compute class probabilities}$$

The spatial filter weights and the classification weights are trained jointly.

Open Questions

- How does the filterbank net compare to FBCSP?
- What can more generic architectures look like?

# 4 NEURAL NETWORK ARCHITECTURES FOR EEG DECODING

Three progressively more generic architectures

- Shallow ConvNet learns temporal filters and later average-pools over large timeregions
- Deep ConvNet uses smaller temporal filters and max-pooling over small timeregions
- Residual network uses many layers with even smaller temporal filters

We continued developing our neural network architectures with our development strategy of progressing from networks that resemble feature-based EEG decopding algorithms to more generic network architectures. After the filterbank network from the master thesis, we adapted the so-called shallow ConvNet, initally also developed in the same master thesis [76]. The shallow ConvNet still resembles filter bank common spatial patterns, but less closely than the filterbank network. After validating that these initial network architectures perform as well as filter bank common spatial patterns, we progressed to developing and evaluating substantially more generic architectures, namely the deep ConvNet and the residual ConvNet.

In this section, I describe the architectures presented in our first publication on EEG deep learning decoding [77]. This part uses text and figures from [77] adapted for readibility in this thesis. The deep and residual ConvNet were primarily developed by me together with help from my coauthors.

## 4.1 SHALLOW CONVNET ARCHITECTURE

We developed the shallow ConvNet architecture, a more flexible architecture than the filterbank network that also learns temporal filters on the input signal and on the later representation. Instead of bandpass-filtered signals, it is fed the raw signals as input. The steps the architecture implements are as follows (also see Figure 4.1):

1. **Temporal Filtering** Learnable temporal filters are indepedently convolved with the signals of each EEG electrode. Afterwards, the channel dimension of the network representation contains electrodes · temporal filters channels.

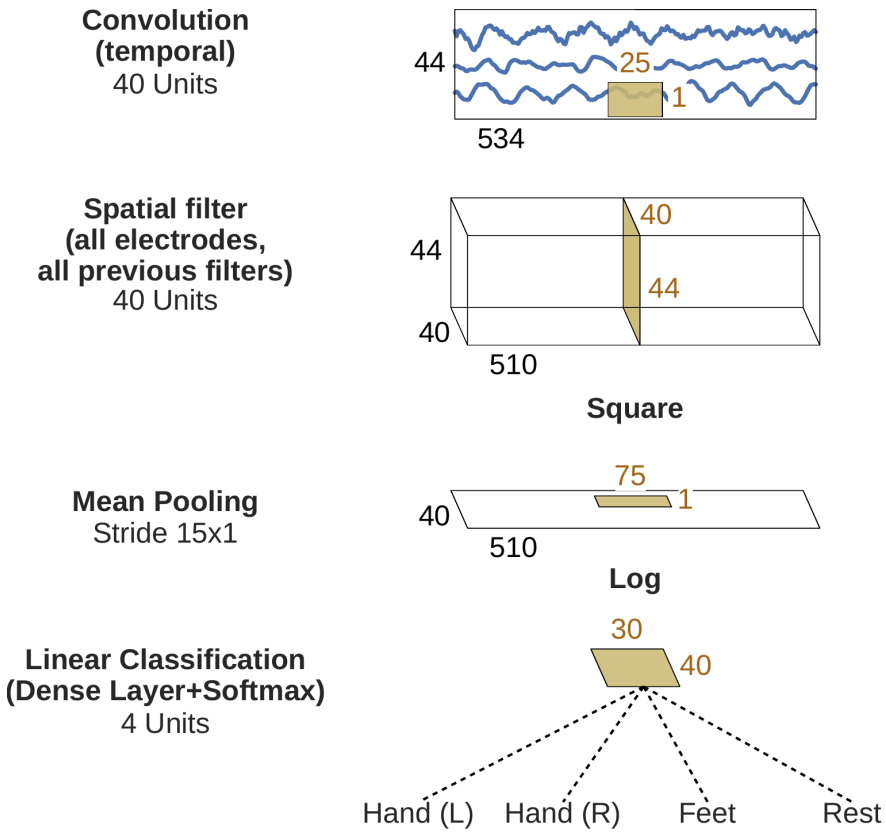



**Figure 4.1: Shallow ConvNet architecture** EEG input (at the top) is progressively transformed toward the bottom, until the final classifier output. Black cuboids: inputs/feature maps; brown cuboids: convolution/pooling kernels. The corresponding sizes are indicated in black and brown, respectively. Note that the final dense layer operates only on a small remaining temporal dimension, making it similar to a regular convolutional layer. Figure from [77].

2. **Spatial Filtering** Combining spatial filtering with mixing the outputs of the temporal filters, the network-channel dimension is linearly transformed by learned weights to a smaller dimensionality for further preprocessing.

3. **Log Average Power** The resulting feature timeseries are then squared, average-pooled and log-transformed, which allows the network to more easily learn log-power-based features. Unlike the filterbank network, the average pooling does not collapse the feature timeseries into one value per trial. So after these processing steps, still some temporal information about the timecourse of the variance throughout the trial can be preserved.

4. **Classifier** The final classification layer transforms these feature timecourses into class probabilities using a linear transformation and a softmax function.

## 4.2 DEEP CONVNET ARCHITECTURE

The deep ConvNet is a more generic architecture, closer to network architectures used in computer vision, see Figure 4.2 for a schematic overview. The first two temporal convolution and spatial filtering layers are the same in the shallow network, which is followed by a ELU nonlinearity (ELUs, $f(x) = x$ for $x > 0$ and $f(x) = e^x - 1$ for $x <= 0$) [17]) and max pooling. The following three blocks simply consist of a convolution, a ELU nonlinearity and a max pooling. In the end, there is again a final linear classification layer with a softmax function. Due to its less specific and more generic computational steps, the deep architecture should be able to capture a large variety of features. Hence, the learned features may also be less biased towards the amplitude features commonly used in task-related EEG decoding.

## 4.3 RESIDUAL CONVNET ARCHITECTURE

We also developed a residual ConvNet ([32]) for EEG decoding. Residual networks add the input of a residual computational block back to its output, and this allows to stably train much deeper networks. We use the same residual blocks as the original paper, described in Figure Figure 4.3. Our residual ConvNet used ELU activation functions 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.

In total, the residual ConvNet has 31 convolutional layers, a depth where ConvNets without residual blocks started to show problems converging in the original residual networks paper [32]. 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 [32]. The overall architecture is also shown in Table 4.1.

Open Questions

- How do these three architectures compare against each other on movement-related decoding tasks?

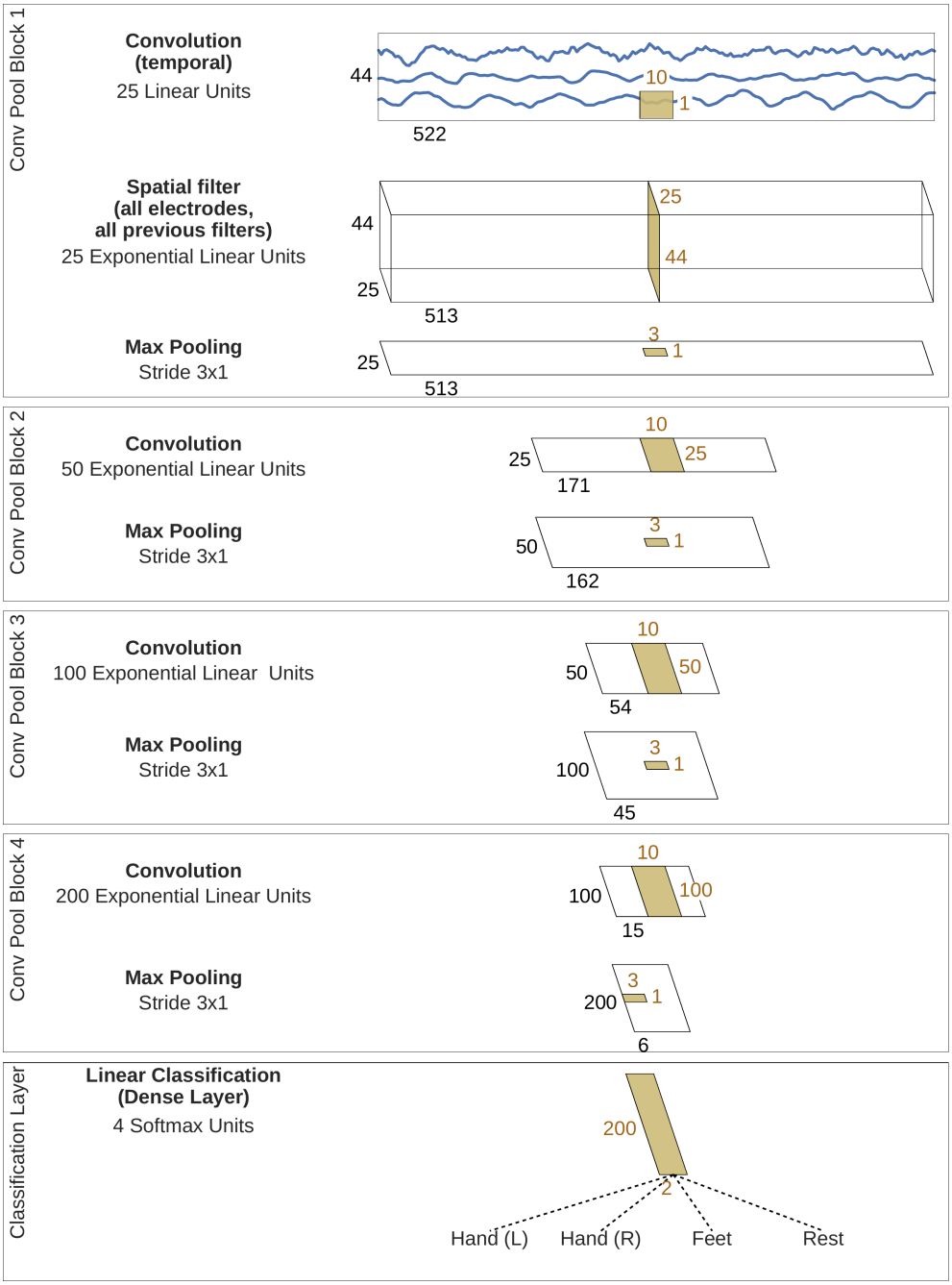



**Figure 4.2: Deep ConvNet architecture.**. Conventions as in Figure 4.1. Figure from [77]

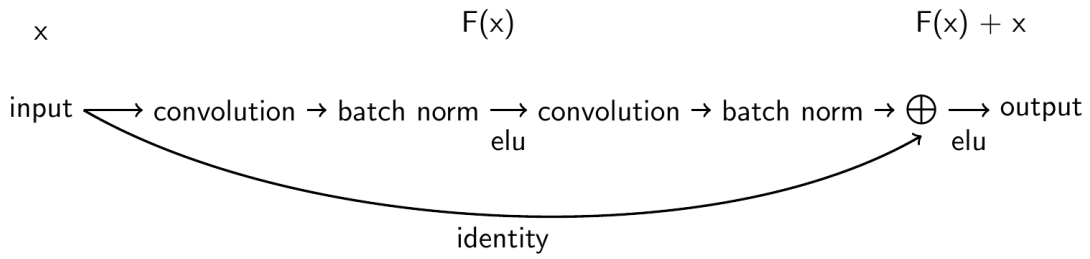



**Figure 4.3: Residual block.** Residual block used in the ResNet architecture and as described in original paper ([32]; see Fig. 2) with identity shortcut option A, except using ELU instead of ReLU nonlinearities. Figure from [77].

| LAYER/BLOCK | #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 4.1: Residual ConvNet architecture hyperparameters.** Number of kernels, kernel and output size for all subparts of the network. Output size is always time x height x channels. Assuming four output classes. 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 the following chapter, we even had 424 predictions, and used the mean of these to predict the trial.

# 5 CROPPED TRAINING

Cropped training means training on many temporal windows within one input example

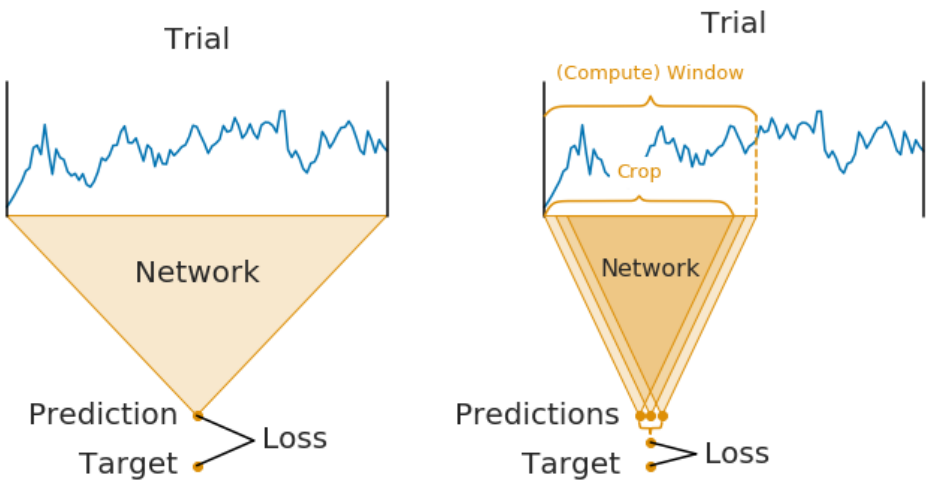
- Greatly increases the number of training examples
- Can be made computationally efficient by avoiding redundant computations

In this chapter, we describe a training strategy called "cropped training" which addresses the problem of the relatively low number of training examples in typical EEG datasets. The goal of this strategy is to improve the performance of deep networks by training them on many sliding temporal windows within the data. This approach had been similarly used as spatial cropping in computer vision, where networks are trained on multiple cropped versions of images. We first describe the concept of regular, non-cropped training and then introduce cropped training on a conceptual level. Finally, we discuss how to implement this approach efficiently. Our aim is to demonstrate the effectiveness and computational efficiency of cropped training as a regularization technique for deep networks on EEG data.

Cropped training for EEG decoding was developed by me in the context of this thesis. Some text and figures are adapted from from [77].

## 5.1 NON-CROPPED/TRIALWISE TRAINING

In the trialwise training of neural networks on EEG data, each example consists of the EEG signals from a single trial and its corresponding label (see Figure 5.1a). This might for example be a 4-second-trial where a subject moved the left hand, with the 4-second-signal as the input and the left hand as a label. Due to the typically small size of EEG datasets, networks trained in this way may only be trained on a few hundred to a few thousand examples per subject. This is significantly fewer examples than those used to train networks in computer vision, where tens of thousands or even millions of images are commonly used.




**(a) Trialwise training.** An entire single trial is fed through the network and the network's prediction is compared to the trial target to train the network.

**(b) Cropped training.** A compute window contains many temporal windows (crops) inside that are used as individual examples to train the network.

**Figure 5.1: Trialwise and cropped training.**

## 5.2 CROPPED TRAINING

Cropped training increases the number of training examples by training on many crops, i.e., temporal windows, within the trial (see Figure 5.1b). For example, in a 4-second trial, all possible 2-second windows within the trial could be used as "independent" examples. This approach drastically increases the number of training examples, although many of the examples are highly overlapping. This can be seen as an extreme version of using random crops of images which is a method used to train deep networks in computer vision. A naive implementation of cropped training would greatly increase the computational cost per epoch due to the highly increased number of examples. Thankfully, the high overlap between neighbouring crops can be exploited for a more efficient implementation.

## 5.3 COMPUTATIONALLY FASTER CROPPED TRAINING

Cropped training can be implemented with substantially less computations by exploiting that highly overlapping crops result in highly overlapping intermediate network activations. By passing a group of neighbouring crops together to the network, we can reuse intermediate computations. See Figure 5.2 and Figure 5.3 for a concrete

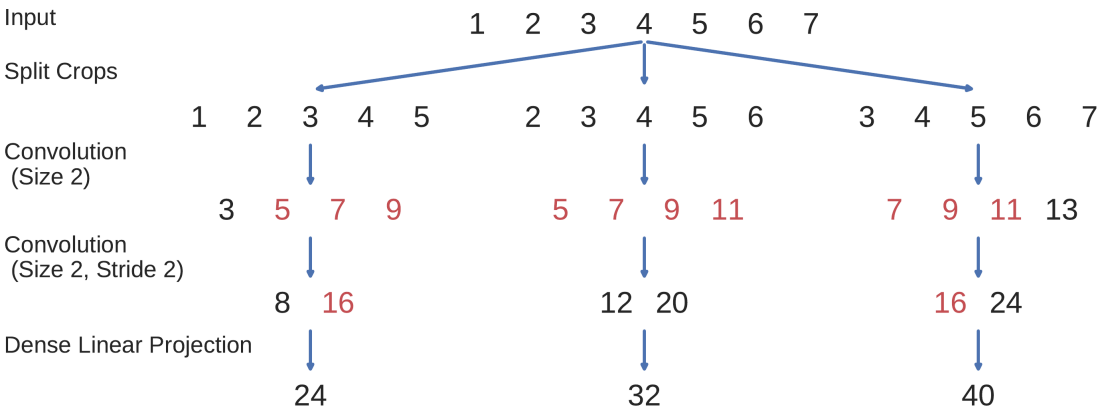



**Figure 5.2: Naive cropped training toy example.** Each possible length-5 crop is taken from the original length-7 trial and independently processed by the Conv-Conv-Linear projection network. All filter values of the network are assumed to be ones. Each crop is processed independently. The values in red are identical and unnecessarily computed independently for each crop. Figure from [77].

example of this speedup method. This idea had been used in the same way for dense predictions on images, e.g., for segmentation [26, 58, 79, 81].

Efficient cropped training then results in the exact same predictions and training as if the neighbouring crops were passed separately through the network. This is only true for networks that either use left-padding or no padding at all to the input and the intermediate activations. In the deep and shallow network described here, we do not use any padding. In the residual network, we use padding, hence the training is not exactly identical to passing neighbouring crops separately, but we still found it to improve over trial-wise training.

The more efficient way to do cropped training introduces a new hyperparameter, the number of neighbouring crops that are decoded together. The larger this hyperparameter, the more computations are saved and the more speedup one gets (see Giusti et al. [26] for a more detailed speedup analysis on images). Larger numbers of neighbouring crops to simultaneously train on require more memory and may also affect the training dynamics due to more neighbouring crops being in the same mini-batch. However, we did not find negative effects on the training dynamics from larger number of simultaneously decoded neighbouring crops, consistent with prior work in computer vision [81].

Open Questions

- For which datasets and architectures does cropped training help or hurt?

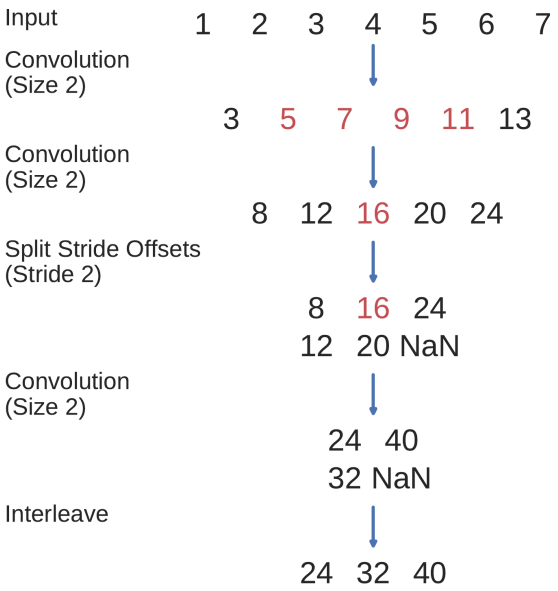



**Figure 5.3: Efficient cropped training.** Each possible length-5 crop is taken from the original length-7 trial and processed simultaneously by the Conv-Conv-Linear projection network, whilce still yielding the same results as if processed independently (Figure 5.3). All filter values of the network are assumed to be ones. Figure from [77].

# 6 PERTURBATION VISUALIZATION

Perturbation visualization perturbs spectral features and measures change in classification predictions

- Can be used to investigate well-known spectral power features
- Can also be implemented through gradients of spectral power features
- Can be extended to investigate phase features

What features the EEG-decoding ConvNet learns is a scientifically interesting question and not straightforward to answer. Through end-to-end training, the networks may learn a variety of features, including brain-signal features and non-brain-signal features, e.g., eye movements that correlate to a hand movement. The learned features may be already known from prior research on brain-signal decoding or represent novel features that had not been described in the literature. However, there is no straightforward way to find out what the deep networks have learned from the brain signals.

Therefore, we developed an input amplitude perturbation method to investigate whether the deep networks learn to extract spectral amplitude features, which are very commonly used in many EEG decoding pipelines. For example, it is known that the amplitudes, for example of the alpha, beta and gamma bands, provide class-discriminative information for motor tasks [5, 60, 61]. Hence, it seems a priori very likely that the deep networks learn to extract such features and worthwhile to check whether they indeed do so. We also extended this method to investigate the use of phase information by the networks.

The amplitude perturbation method was developed by me in the context of this thesis, the phase perturbation method was developed by Kay Hartmann together with me. Text and figures in this chapter are adapted from [77] and [30].

## 6.1 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 6.1 for an overview). Concretely, the visualization method performs the following steps:

1. Transform all training trials into the frequency domain by a Fourier transformation
2. Randomly perturb the amplitudes by adding Gaussian noise (with mean 0 and variance 1) to them (phases were kept unperturbed)
3. Retransform perturbed trials to the time domain by the inverse Fourier transformation
4. Compute 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)
5. Repeat this procedure with 400 perturbations sampled from aforementioned Gaussian distribution
6. Correlate 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 [89]. 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.

This method can not only be applied to final predictions, but also to investigate any intermediate network filter's activations in order to better understand the intermediate computations of the network.

## 6.2 GRADIENT-BASED IMPLEMENTATION

An simpler and more computationally efficient way to implement the idea of testing the sensitivity of the network to spectral amplitude features is through gradient-based analysis [1]. There, we directly compute the gradient of the output unit with respect to the amplitudes of all frequency bins of all electrodes of the original unperturbed trial. To practically implement this, one must first transform the time domain input signal into the frequency domain and to a amplitude/phase representation via the Fourier transform. Then, one can transform the amplitude/phase representation back

1 This idea was suggested to us in personal communication by Klaus-Robert Müller.

to the time domain using the inverse Fourier Transform. Since the inverse Fourier transform is differentiable, one can backpropagate the gradients from the output unit through the time domain input back to the amplitudes. The more the network behaves locally linear around the input, the closer the results of this variant would be to the original perturbation variant. It is computationally substantially faster as everything is computed in one forward-backward pass, without needing to iterate over many perturbations. The gradient-based method may result in less insightful visualizations if the prediction function of the network is locally very nonlinear at a given pint but has an approximately linear relationship with the spectral amplitudes in a larger neighbourhood around that point. See works on other saliency/gradient-based visualizations for discussions in this topic, e.g. Sturmfels, Lundberg, and Lee [86].

## 6.3 EXTENSION TO PHASE-BASED PERTURBATIONS

The amplitude-perturbation method can also be extended to investigate in how far networks are affected by changes in phase features. The response of filters to changes in the phase of certain frequencies was calculated similarly to the amplitude perturbation correlations. However, because of the cyclic nature of phase features, the change of activations in a filter resulting from a phase shift cannot be quantified using the mean activation difference. When you change the phase of the frequency that a filter is sensitive to, you won't expect the activations of the filter to increase uniformly throughout the window. Instead, the activations of a phase-sensitive filter will probably be temporally shifted by the phase change. Units of filters whose receptive field contained its specific phase in the original signal should activate less and units whose receptive field contains the specific phase in the perturbed signal should then activate more. Therefore, the original activations and the activations on the perturbed input should have a decreased correlation (less than 1). Activation and correlation should remain similar for phase-insensitive filters.

Phase perturbations were sampled from $p^P_{\xi,i} \sim N(0, \pi)$. Perturbed phases $P^P_\xi$ were calculated by shifting the phase: $P^P_\xi(X_i) = p^P_{\xi,i} + P_\xi(X_i)$. Perturbed signals $X^P$ were reconstructed by inverse Fourier transformation. The correlation between original and perturbation filter activations of a filter $f$ from trial $i$ is denoted by $\rho_{y_{f,i}, y^P_{f,i}} = corr(y_{f,i}, y^P_{f,i})$. Correlations between phase perturbations $p^P_\xi$ and filter activity correlations $\rho_{y_f, y^P_f}$ were calculated identically to amplitude perturbations. The resulting mean absolute phase perturbation correlations for each layer is denoted as $\varrho^P_{l,\xi}$.

Since we wanted to study only the effect of changing the overall phase of the signal, independent of the effect of increased or decreased phase synchronicity across EEG

channels, we did not perturb the phase in channels individually, but applied one phase perturbation of a certain frequency in all channels equally.

## 6.4 INTERPRETATION AND LIMITATIONS

The perturbation-based visualization reflects network behavior and one cannot directly draw inferences about the data-generating process from it. This is because a prediction change caused by an amplitude perturbation may reflect both learned task-relevant factors as well as learned noise correlations. For example, increasing the alpha-band amplitude at electrode C4 (located on the right side), may increase the predicted probability for right hand movement. That would likely not be because the alpha amplitude actually increases at C4 during right hand movement, but because the amplitude *decreases* on C3 *and* is correlated between C3 and C4. Hence, first subtracting the C4 amplitude from the C3 amplitude and then decoding associating negative values of this computation with right hand movement is a reasonable learned prediction function. And this learned prediction function would cause the amplitude-perturbation function to show that an alpha increase at C4 causes an increase in the predicted probability for right hand movement. For a more detailed discussion of this effect in the context of linear models, see [31].

Open Questions

- Do spectral maps obtained from this visualization show neurophysiologically plausible patterns?
- What can they reveal about the inner computations of the networks?

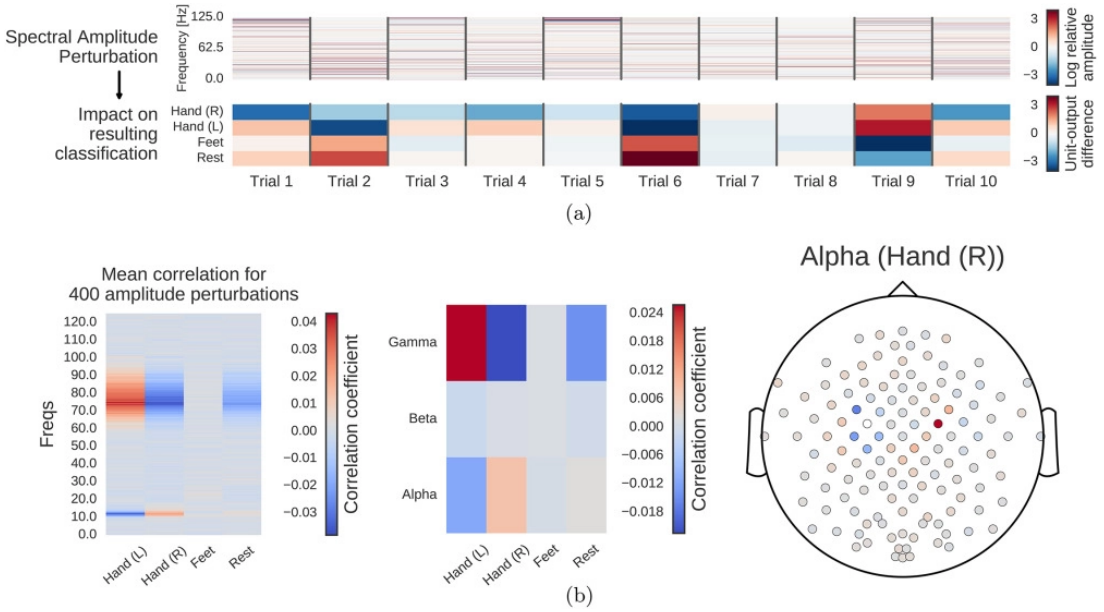



**Figure 6.1:**

**Computation overview for input-perturbation network-prediction correlation map.** (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. Figure from [77].

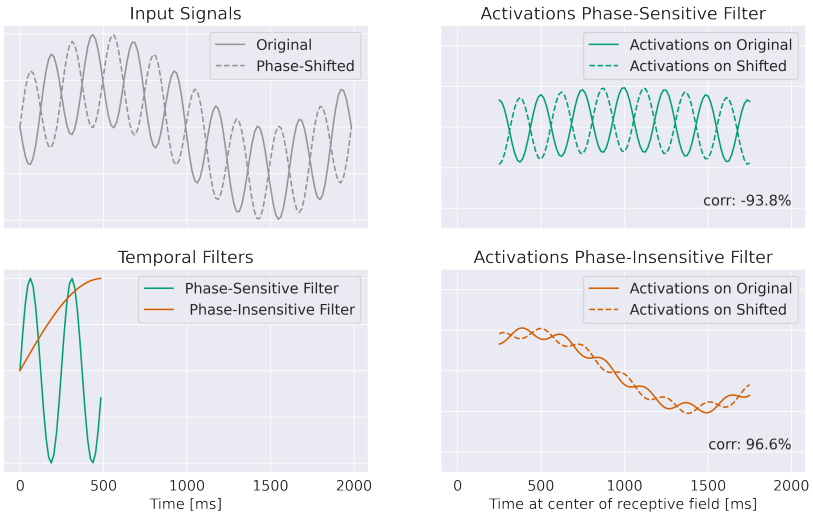



**Figure 6.2:**

**Intuition of the phase perturbation correlation method.** Top left: two input signals where the phase of one frequency has been shifted between the two signals. Bottom left: two temporal filters, one phase-sensitive and one phase-insensitive to the frequency where the phase was shifted. Right: activations for the phase-sensitive and phase-insensitive filter for the original and the shifted signal. The correlation between activations of the phase-sensitive filter is very low, even negative (-93.8%), whereas correlation between the activations of the phase-insensitive filter remains high at 96.6%. Note that this simplified example does not illustrate some key mechanisms for activations to become more generally phase-insensitive such as nonlinearities and pooling operations.

# 7 INVERTIBLE NETWORKS

Invertible networks can help understand learned discriminative features in the EEG

- Class prototypes can be visualized
- Per-electrode prototypes may be even more interpretable
- Additionally, a small interpretable network may allow further insights

Invertible networks are networks that are invertible by design, i.e., any network output can be mapped back to a corresponding input bijectively [22, 23, 34, 40, 69]. The ability to invert any output back to the input enables different interpretability methods and furthermore allows training invertible networks as generative models via maximum likelihood.

This chapter starts by explaining invertible layers that are used to design invertible networks, proceeds to detail training methodologies for invertible networks as generative models or classifiers, and goes on to outline interpretability techniques that help reveal the learned features crucial for their classification tasks.

## 7.1 INVERTIBLE LAYERS

Invertible networks use layers constructed specifically to maintain invertibility, thereby rendering the entire network structure invertible. Often-used invertible layers are coupling layers, invertible linear layers and activation normalization layers [40].

**Coupling layers** split a multidimensional input $x$ into two parts $x_1$ and $x_2$ with disjoint dimensions and then use $x_2$ to compute an invertible transformation for $x_1$. Concretely, for an additive coupling layer, the forward computation is:

$$y_1 = x_1 + f(x_2) \qquad \text{Compute } y_1 \text{ from } x_1 \text{ and arbitrary function f of } x_2$$
$$y_2 = x_2 \qquad \text{Leave } x_2 \text{ unchanged}$$

The inverse computation is:

$$x_1 = y_1 - f(y_2) \qquad \text{Invert to } x_1 \text{ using unchanged } y_2 = x_2$$
$$x_2 = y_2 \qquad x_2 \text{ was unchanged}$$

For the splitting of the dimensions in a timeseries, there are multiple ways, such as using the even time indices as $x_1$ and all the odd time indices as $x_2$ or using difference and mean between two neighbouring samples (akin to one stage of a Haar Wavelet). The function $f$ is usually implemented by a neural network, in our cases it will be small convolutional networks. Instead of addition any other invertible function can be used, affine transformation are commonly used, where $f$ produces translation and scaling coefficients $f_t$ and $f_s$:

$$y_1 = x_1 \cdot f_s(x_2) + f_t(x_2) \qquad y_2 = x_2 \qquad \text{Affine Forward}$$

$$x_1 = \frac{(y_1 - f_t(y_2))}{f_s(y_2)} \qquad x_2 = y_2 \qquad \text{Affine Inverse}$$

**Invertible linear layers** compute an invertible linear transformation (an automorphism) of their input. Concretely they multiply a $d$-dimensional vector $\mathbf{x}$ with a $dxd$-dimensional matrix $W$, where $W$ has to be invertible, i.e., have nonzero determinant.

$$y = W\mathbf{x} \qquad \text{Linear Forward}$$
$$x = W^{-1}\mathbf{y} \qquad \text{Linear Inverse}$$

For multidimensional arrays like feature maps in a convolutional network, these linear transformations are usually done per-position, as so-called invertible 1x1 convolutions in the 2d case.

**Activation normalization layers** perform an affine transformation with learned parameters with $s$ and $t$ learned scaling and translation parameters (independent of the input $x$):

$$y = x \cdot s + t \qquad \text{ActNorm Forward}$$
$$x = \frac{y - t}{s} \qquad \text{ActNorm Inverse}$$

These have also been used to replace batch normalization and are often initialized data-dependently to have standard-normalized activations at the beginning of training.

## 7.2 GENERATIVE MODELS BY MAXIMIZING AVERAGE LOG LIKELIHOOD

Invertible networks can also be trained as generative models via maximizing the average log likelihood. In this training, the network is optimized to maximize the average log probabilities of the training inputs, which is equivalent to minimizing the Kullback-Leibler (KL) divergence between the training distribution and the learned model distribution [92]. Invertible networks assign probabilities to training inputs $x$ by mapping them to a latent space $z = f(x)$ and computing their probabilities under a predefined prior $p_z(z)$ in that latent space. For real-valued inputs, one has to account for quantization and volume change to ensure this results in a proper probability distribution $p_x$ in the input space. Quantization refers to the fact that training data often consists of quantized measurements of underlying continuous data, e.g. digital images can only represent a distinct set of color values. Volume change refers to how the invertible networks' mapping function $f$ expands or squeezes volume from input space to latent space.

### 7.2.1 (De)quantization

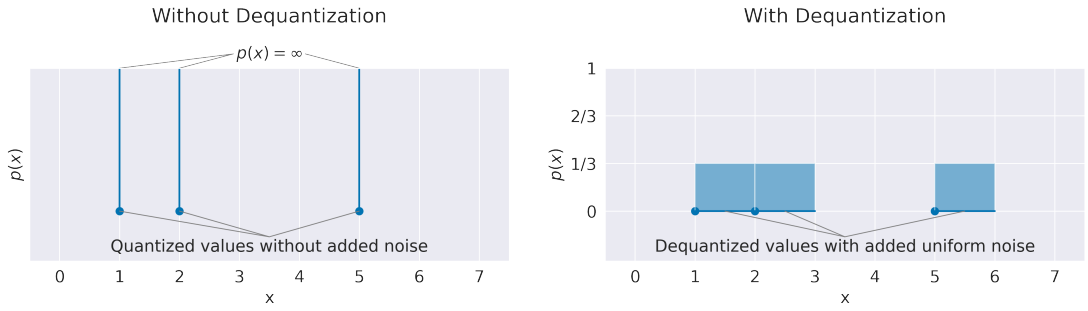



**Figure 7.1: Average log-likelihood-maximizing distributions without and with dequantization.** Examples show result of fitting quantized values like discrete integer color values with a continuous probability distribution. Example training distributions have 3 data points at $x_1 = 1$, $x_2 = 2$ and $x_3 = 5$. On the left, fitting quantized values directly leads to a pathological solution as the learned distribution $p$ can assign arbitrarily high probability densities on the data points. On the right, adding uniform noise $U(0, 1)$ to the datapoints leads to a distribution that also recovers the correct discrete distribution, that means integrating over the probability densities in the volume of each point leads to $P(x_i) = \frac{1}{3}$.

Often, training data for neural networks consists of quantized measurements like discrete integer color values from 0 to 255, which are mapped to real-world floating point numbers for training. Naively maximizing the average log probability densities

of these quantized values with a continuous probability distribution would lead to pathological behavior as the quantized training data points do not cover any volume. Hence it would be possible for the learned distribution to assign infinitely high probability densities to individual data points, see Figure 7.1 for an illustration.

Hence, one needs to "dequantize" the data such that each datapoint occupies volume in the input space [23, 34]. The simplest way here is to add uniform noise to each data point with a volume corresponding to the gap between two data points. For example, if the 256 color values are mapped to 256 floating values between 0 and 1, one may add uniform noise $u \sim (0, \frac{1}{256})$ to the inputs. Then the KL-divergence between the dequantized continuous distribution and the learned continuous distribution is upper bounded by the KL-divergence between the underlying discrete distribution and the learned discrete distribution obtained by integrating over the noise samples for each input [92]. Since in our case, we are not primarily interested in the exact performance as a generative model in terms of number of bits, we simply add gaussian noise with a fixed small standard deviation $N(0, 0.005I)$ during training.

### 7.2.2 Volume Change

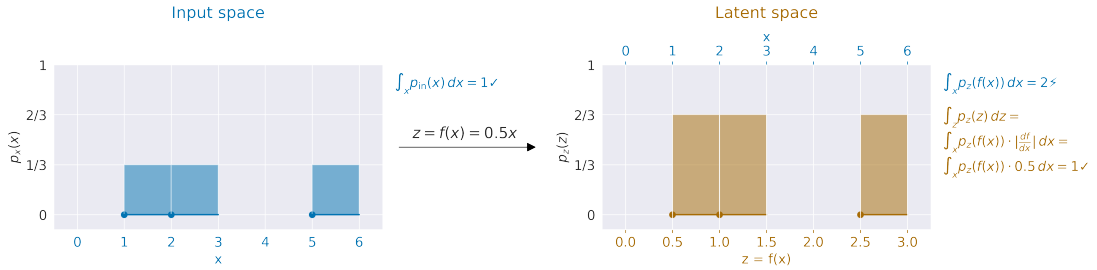



**Figure 7.2: Probability densities and volume changes.** Computing probability densities taking into account how a mapping function changes volume. Input $x$ with probability distribution $p_x(x)$ on the left is scaled by 0.5 to $z = f(x) = 0.5x$ with probability distribution $p_z(z)$ on the right. Naively integrating $p_z(f(x))$ over x would lead to a non-valid probability distribution with $\int_x p_z(f(x))\,dx = 2$. To get the prober probability densities in input space from $p_z(z)$, one has to multiply with the volume changes, in this case the scaling factor of the mapping $f(x)$ from x to z, giving $p_x(x) = p_z(f(x)) \cdot \frac{df}{dx} = p_z(f(x)) \cdot 0.5$ which correctly integrates to 1.

In addition, for these probability densities in latent space to form a valid probability distribution in the input space, one has to account for how much the network's mapping function squeezes and expands volume. Otherwise, the network can increase densities by squeezing all the inputs closely together in latent space, see also Figure 7.2 for a onedimensional example. To correctly account for the volume change during

the forward pass of $f$ one needs to multiply the probability density with the volume change of $f$, descreasing the densities if the volume is squeezed from input to latent space and increasing it if the volume is expanded. As the volume change at a given point $x$ is given by the absolute determinant of the jacobian of f at that point $\det\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)$, the overall formula looks like this:

$$p(x) = p_z(f(x)) \cdot |\det\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)| \tag{7.1}$$

Or in log-densities:

$$\log p(x) = \log p_z(f(x)) + \log |\det\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)| \tag{7.2}$$

## 7.3 GENERATIVE CLASSIFIERS

Invertible networks trained as class-conditional generative models can also be used as classifiers. Class-conditional generative networks may be implemented in different ways, for example with a separate prior in latent space for each class. Given the class-conditional probability densities $p(x|c_i)$, one can obtain class probabilities via Bayes' theorem as $p(c_i|x) = \frac{p(x|c_i)}{\sum_j p(x|c_j)}$.

Pure class-conditional generative training may yield networks that perform badly as classifiers. One proposed reason is the relatively small reduction in optimal average log likelihood loss obtainable from providing the class label to the network for high-dimensional inputs, often much smaller than typical differences between two runs of the same network [92]. The reduction in the optimal average log likelihood loss through providing the class label can be derived from a compression perspective. According to Shannon's theorem, more probable inputs need less bits to encode than less probable inputs, or more precisely Number of bits needed$(x) = \log_2 p(x)$. How many of these bits are needed for the class label in case it is not given? To distinguish between n classes, one needs only $\log_2(n)$ bits, so in case of binary pathology classification, only 1 bit is needed. Therefore the optimal class-conditional model will only be 1 bit better than the optimal class-independent model. However, the inputs themselves typically need at least 1 bit per dimension, so already, a 21 channel x 128 timepoints EEG-signal may need at least 2688 bits to encode. Hence, the class encoding contributes very little to the overall encoding size and maximum likelihood loss. In contrast, the loss difference between two training runs of the same network will typically be at least one to two orders of magnitude larger. Still, in practice, the gains from using a class-conditional model, by e.g., using a separate prior per class in latent space, are often larger, but it is not a priori clear if the reductions in

loss from exploiting the class label are high enough to result in a good classification model.

Various methods have been proposed to improve the performance of using generative classifiers. For example, people have fixed the per-class latent gaussian priors so that they retain the same distance throughout training [37] or added a classification loss term

$$\begin{aligned} L_{\text{class}}(x, c_i) &= -\log p(c_i|x) \\ &= -\log \frac{p(x|ci)}{\sum_j p(x|cj)} \\ &= -\log \frac{e^{\log p(x|ci)}}{\sum_j e^{\log p(x|cj)}} \\ &= -\log \left(\text{softmax}\left(\log p(x|c_i)\right)\right) \end{aligned}$$

to the training loss[4]. In our work, we experimented with adding a classification loss term to the training, and also found using a learned temperature before the softmax helps the training, so leading to:

$$L_{\text{class}}(x, c_i, t) = -\log \frac{e^{\frac{\log p(x|ci)}{t}}}{\sum_j e^{\frac{\log p(x|cj)}{t}}} = -\log \left(\text{softmax}\left(\frac{\log p(x|c_i)}{t}\right)\right) \tag{7.3}$$

$$\tag{7.4}$$

Our overall training loss is simply a weighted sum of generative loss and classification loss:

$$L(x, c_i, t) = L_{\text{class}}(x, c_i, t) + L_{\text{gen}}(x, c_i) \tag{7.5}$$

$$= -\log \left(\text{softmax}\left(\frac{\log p(x|c_i)}{t}\right)\right) - \alpha \log p(x|ci) \tag{7.6}$$

$$\tag{7.7}$$

where we choose the hyperparameter $\alpha$ as the inverse of the number of dimensions $\alpha = \frac{1}{\text{Number of dimensions of x}}$.

## 7.4 INVERTIBLE NETWORK FOR EEG DECODING

We designed an invertible network named EEG-InvNet for EEG Decoding using invertible components used in the literature, primarily from the Glow architecture

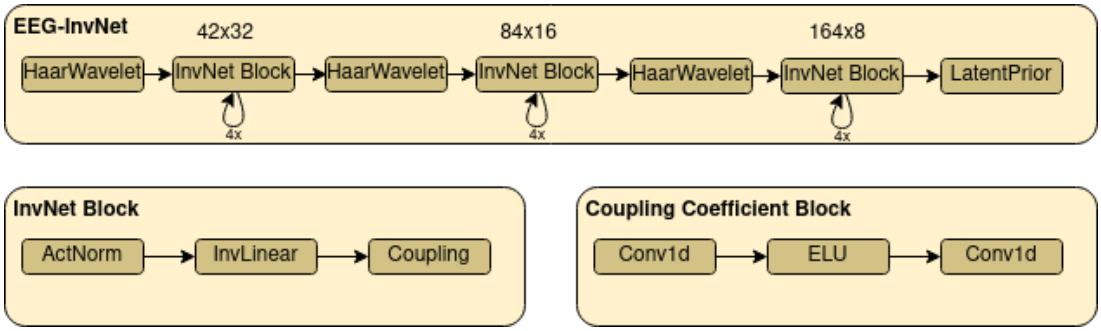



**Figure 7.3: EEG-InvNet architecture.** Our EEG-InvNet architecture consists of three stages that operate at sequentially lower temporal resolutions. Input is two seconds of 21 electrodes at 64 Hz so 21x128 dimensions. These are downsampled using Haar Wavelets to 42x32 for the first, 84x16 for the second and 164x8 for the last stage. One stage consists of 4 blocks, each block has an activation normalization, an invertible linear and a coupling layer. The activation normalization and invertible linear layer act on the channel dimension, so perform the same operation across channels on timepoint in the feature map. The coupling layer uses two convolutions with an exponential linear unit activation inbetween.

[40]. Our architecture consists of three stages that operate on sequentially lower temporal resolutions. Similar to Glow, the individual stages consists of several blocks of Activation Normalization, Invertible Linear Channel Transformations and Coupling Layers, see Figure 7.3. Between each stage, we downsample by computing the mean and difference of two neighbouring timepoints and moving these into the channel dimension. Unlike Glow, we keep processing all dimensions throughout all stages, finding this architecture to reach competitive accuracy on pathology decoding. We use one gaussian distribution per class in the latent space. We experimented with affine and additive coupling layers, and report results for additive layers as the restricted expressiveness may make them easier to interpret.

## 7.5 CLASS PROTOTYPES

In our first visualization, we show the inputs resulting from inverting the means of the gaussian distributions for each class (see Figure 7.4) . These can be seen as prototypical examples of each class and may give hint about the discriminative features that have been learned. As these are only single examples, they need to be interpreted cautiously. For example, individual features within the examples may have a variety of relationships with the actual prediction function. Consider if a prototype contains a large alpha-band oscillation at two electrodes, then these may be indepedendently predictive predictive of that class or only in combination or

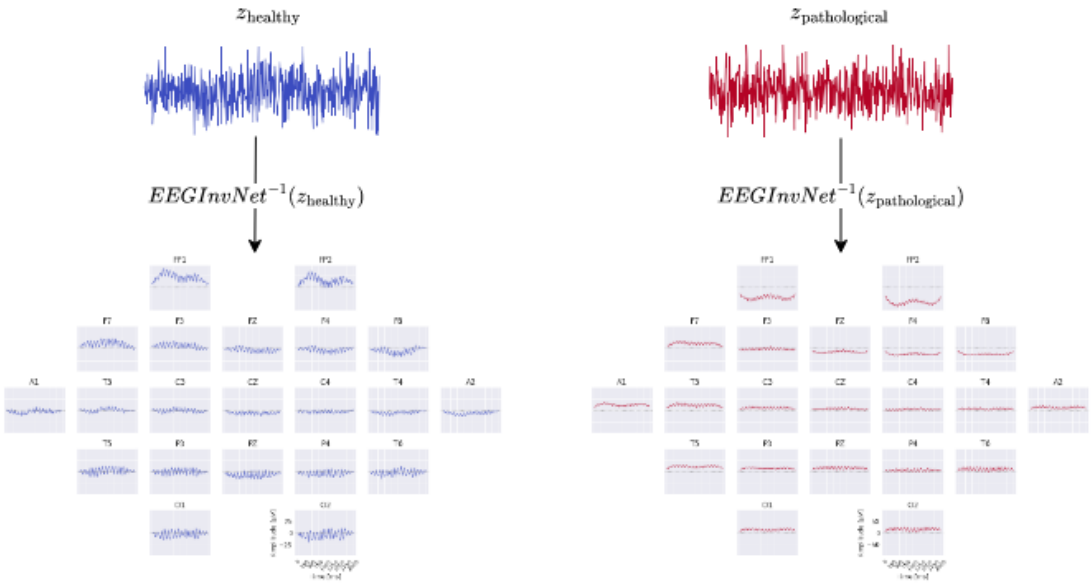



**Figure 7.4: EEG-InvNet class prototypes.** Class prototypes are synthesized by inverting the means $z_{\text{healthy}}$ and $z_{\text{pathological}}$ of the per-class gaussian distributions.

even only in some combination with other features. Nevertheless, the prototypes can already suggest potential discriminative features for further investigation.

## 7.6 PER-ELECTRODE PROTOTYPES

One way to get more interpretable prototypes is to synthesize them per electrode. Here, we synthesize a signal $x^*_{e_k}$ for a specific electrode $e_k$ such that the class prediction is high for one class, independent of the signals at the other electrodes (see Figure 7.5). So for electrode $e_k$ and class $c_i$, we aim to optimize the signal $x^*_{e_k}$ by maximizing the marginals $p(x^*_{e_k}|c_i) = \int p(x|c_i; x_{e_k} = x^*_{e_k})dx$ (generative loss) and $p(c_i|x^*_{e_k}) = \frac{p(x^*_{e_k}|c_i)}{\sum_j p(x^*_{e_k}|c_j)}$ (classification loss). To approximate this, we sample $n$ signals $x_l$ of the training distribution and replace the signal $x_{l,e_k}$ of the electrode $e_k$ we are synthesizing by the optimized $x^*_{e_k}$. This leads to $p(x^*_{e_k}|c_i) \approx \sum_{l=1}^{n} p(x_l|c_i; x_{l,e_k} = x^*_{e_k})$. While being only a coarse approximation, this already yields insightful visualizations. For the classification loss, when computing $p(c_i|x^*_{e_k})$, we found it helpful to first divide the log probabilities $\log p(x_l|c_i; x_{l,e_k} = x^*_{e_k})$ by the learned temperature $t$ of the classifier: $\log p_{\text{clf}}(x^*_{e_k}|c_i) = \text{logsumexp}\left(\frac{\log p(x_l|c_i; x_{l,e_k}=x^*_{e_k})}{t}\right)$. Otherwise, $\sum_n p(x_n|c_i; x_{n,e_k} = x^*_{e_k})$ may be dominated by just a few samples when computing $p(c_i|x^*_{e_k})$. We only apply this for the classification loss $p(c_i|x^*_{e_k})$, not the generative loss $p(x^*_{e_k}|c_i)$.


**Figure 7.5: EEG-InvNet per-electrode class prototypes.** For getting per-electrode prototypes, class-specific signals for one electrode are synthesized while signals at other electrodes are sampled from training data. In the example, prototypes for T3 for the healthy and pathological class are learned, four samples for remaining electrodes are shown. In practice, a much larger number of samples would be used. Class signal probabilities are marginalized over the non-optimized channels as explained in text.

## 7.7 EEG-COSNET

Finally, we also implemented a small convolutional network EEG-CosNet that we designed to be directly interpretable. We tried to distill the trained EEG-InvNet into the EEG-CosNet by training the EEG-CosNet using the EEG-InvNet class probabilities as the targets for the classification loss $L_{\text{class}}$. Our EEG-CosNet consists of just three steps (see Figure 7.6 for an example computation):

### Spatial Filtering

$$h_1 = W_s^T x \qquad \text{Apply spatial filter weights } W_s \text{ to inputs}$$

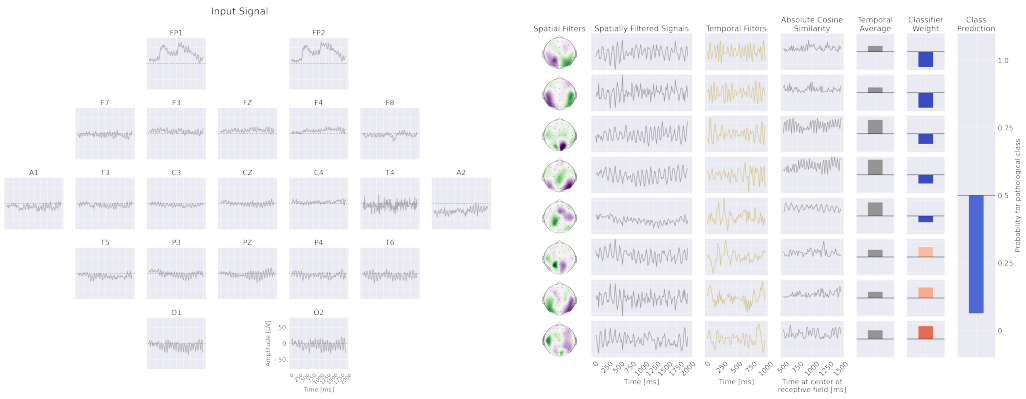



**Figure 7.6: Example processing of the EEG-CosNet.** Example EEG input on the left, then processing steps on the right: spatial filtering, absolute cosine similarity with temporal filters, temporal averaging, then weighting with linear classifier weights for class prediction. Note the EEG-CosNet in this visualization only uses 8 filters, whereas later we will use 64 filters.

### Absolute Moving Cosine Similarity with Temporal Filters

$$h_2 = |\text{cos\_sim}(h_1, \text{filters})| \quad \text{Absolute moving cosine similarity with temporal filters}$$

$$h_3 = \frac{\sum_t (h_2)}{n_{\text{times}}} \quad \text{Average over timepoints in trial}$$

### Classification

$$h_4 = W_c^T h_3 \quad \text{Apply classifierweights } W_c \text{ on these features}$$

$$p(\text{c}_{\text{path}}|h_4) = \frac{1}{1 + \sum_j e^- h_4} \quad \text{Compute sigmoid to get pathological probability}$$

Steps 1 and 2 yield spatiotemporal patterns that can be visualized as waveforms and scalp plots, and that are weighted by the linear classifier for the respective classes. We chose cosine similarity to ensure that high output values correspond to spatially filtered signals that resemble the corresponding temporal filter. The spatial filter weights and linear classifier weights can be made even more interpretable through transforming the discriminative weights into generative patterns by multiplying them with the covariance of the electrodes/averaged absolute cosine similarities after training, see Haufe et al. [31] for a discussion on this technique. We use 64 spatiotemporal filters with temporal length 64 corresponding to one second at 64 Hz.

Open Questions

- How well can the EEG-InvNet perform on EEG Pathology decoding?
- What features can the EEG-InvNet reveal?
- How well can the EEG-CosNet approximate the EEG-InvNet?
- What features can the EEG-CosNet reveal?

# 8 DECODING MOVEMENT-RELATED BRAIN ACTIVITY

ConvNets can perform as well as FBCSP on movement-related decoding

- Shallow network performs best on highpassed data
- Deep network strongly benefits from cropped training on non-highpassed data
- Residual network performs well with right initialization
- Methodological improvements from computer vision improve EEG deep learning decoding
- Visualizations reveal use of spectral power features with neurophysiologically plausible patterns

Movement-related decoding problems are among the most researched in EEG decoding and were hence our problem of choice for the first evaluation of deep learning on EEG. A typical movement-related experimental setting is that subjects receive a cue for a specific body part (e.g. right hand, feet, tongue, etc.) and either move (motor execution) or imagine to move (motor imagery) this body part. The EEG signals recorded during the imagined or executed movements then often contain patterns specific to the body part being moved or thought about. These patterns can then be decoded using machine learning. In the following, I will describe our study on movement-related EEG decoding using deep learning, mostly using content adapted from Schirrmeister et al. [77] and from [30].

The main study as published in Schirrmeister et al. [77] was carried out by me as the main author with help of al the coauthors. For Hartmann, Schirrmeister, and Ball [30], Kay Hartmann was the main contributor and I supported and advised him.

## 8.1 DATASETS

### 8.1.1 High-Gamma Dataset

Our High-Gamma Dataset is a 128-electrode dataset (of which we later only use 44 sensors covering the motor cortex) obtained from 14 healthy subjects (6 female, 2 left-handed, age 27.2±3.6 (mean±std)) with roughly 1000 (963.1±150.9, mean±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 [5, 20]. 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 [18, 29, 63]. 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 $\mu V$ RMS 0.16–200 Hz, 1.5 $\mu V$ 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 electromagnetic shielding combined with the comfortable armchair, dimly lit Faraday cabin and the relatively long 3-4 second inter-trial intervals (see below) were used to minimize artifacts produced by the subjects during the trials.

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 [74]. The experiment was approved by the ethical committee of the University of Freiburg.

### 8.1.2 BCI Competition IV 2a

The BCI competition IV dataset 2a 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), for details see Brunner et al. [12]. The training set consists of the 288 trials of the first session, the test set of the 288 trials of the second session.

### 8.1.3 BCI Competition IV 2b

The BCI competition IV dataset 2b is a 3-electrode EEG motor-imagery dataset with 9 subjects and 5 sessions of imagined movements of the left or the right hand, the latest 3 sessions include online feedback [44]. The training set consists of the approx. 400 trials of the first 3 sessions (408.9$\pm$13.7, mean$\pm$std), the test set consists of the approx. 320 trials (315.6$\pm$12.6, mean$\pm$std)of the last two sessions.

## 8.2 PREPROCESSING

We only minimally preprocessed the data to allow the networks to extract as much information as possible while keeping the input distribution in a value range suitable for stable network training.

Concretely, our preprocessing steps were:

1. **Remove outlier trials:** Any trial where at least one channel had a value outside +- 800 mV was removed to ensure stable training.
2. **Channel selection:** For the high-gamma dataset, we selected only the 44 sensors covering the motor cortex for faster and more accurate motor decoding.
3. **High/Bandpass (optional) :** Highpass signal to above 4 Hz. This should partially remove potentially informative eye components from the signal and ensure that the decoding relies more on brain signals. For the BCI competition datasets, in this step we bandpassed to 4-38 Hz as using only frequencies until ~38-40 Hz was commonly done in prior work in this dataset.
4. **Standardization:** Exponential moving standardization as described below to make sure the input distribution value range is suitable for network training.

Our electrode-wise exponential moving standardization computes exponential moving means and variances with a decay factor of 0.999 for each channel and used these to standardize the continuous data. Formally,

$$\mu_t = 0.001 x_t + 0.999 \mu_{t-1} \tag{8.1}$$
$$\sigma_t^2 = 0.001 (x_t - \mu_t)^2 + 0.999 \sigma_{t-1}^2 \tag{8.2}$$
$$x't = (x_t - \mu_t) / \sqrt{\sigma_t^2} \tag{8.3}$$
$$\tag{8.4}$$

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. 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 as we validated later, see Chapter 9.

## 8.3 TRAINING DETAILS

As our optimization method, we used Adam [39] together with a specific early stopping method, as this consistently yielded good accuracy in preliminary experiments on the training set. 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 [39]. The early stopping strategy that we use throughout these experiments, developed in the computer vision field [1], 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.

1 https://web.archive.org/web/20160809230156/https://code.google.com/p/cuda-convnet/wiki/Methodology

Note that in later works we do not use this early stopping method anymore as we found training on the whole training set with a cosine learning rate schedule [47] to lead to better final decoding performance.

## 8.4 DESIGN CHOICES

| 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), as 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 the section "Datasets and Preprocessing"). |
| Pooling mode | Max | Mean | (see above) |
| Regularization and intermediate normalization | Dropout + batch normalization + a new tied loss function (explanations see text) | Only batch normalization, only dropout, neither of both, nor tied loss | We wanted to investigate whether recent deep learning advances improve accuracies and check how much regularization is required. |
| Factorized temporal convolutions | One 10 x 1 convolution per convolutional layer | Two 6 x 1 convolutions per convolutional layer | Factorized convolutions are used by other successful ConvNets [88] |
| Splitted vs one-step convolution | Splitted convolution in first layer (see Section 4.1) | 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 8.1: Evaluated design choices.**

For the shallow and deep network, we evaluated how a number of design choices affect the final accuracies, see Table 8.1.

### 8.4.1 Tied Loss Function

Our tied loss function penalizes the discrepancy between neighbouring predictions. Concretely, in this *tied sample loss function*, we added the cross-entropy of two neighboring predictions to the usual loss of of negative log likelihood of the labels. So, denoting the predicted probabilties $p\big(l_k|f_k(X^j_{t..t+T'};\theta)\big)$ for crop $X^j_{t..t+T'}$ with label $l_k$ from time step $t$ to $t+T'$ by $p_{f,k}(X^j_{t..t+T'})$, the loss now also depends on the predicted probabilties for the next crop $p_{f,k}(X^j_{t..t+T'+1})$ and is then:

$$\begin{aligned}\text{loss}\big(y^j, p_{f,k}(X^j_{t..t+T'})\big) = &\sum_{k=1}^{K} -log\big(p_{f,k}(X^j_{t..t+T'})\big)\cdot\delta(y^j = l_k)\\ &+\sum_{k=1}^{K} -log\big(p_{f,k}(X^j_{t..t+T'})\big)\cdot p_{f,k}(X^j_{t..t+T'+1})\end{aligned} \tag{8.5}$$

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

## 8.5 RESULTS

### 8.5.1 Validation of FBCSP Pipeline

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 [73]. 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 comparing against 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.

### 8.5.2 Filterbank Network

Prior to our more extensive study, we had evaluated the filterbank network on a different version of the High-Gamma Dataset in a master thesis [76]. This version of the dataset included different subjects, as some subjects had not been recorded yet and other subjects were later excluded for the work here due to the presence of too

| DECODING METHOD | SAMPLING RATE | TEST ACCURACY [%] | STD [%] |
|---|---|---|---|
| FBCSP | 300 | 88.1 | 13.9 |
| FBCSP | 150 | 86.7 | 14.3 |
| Filterbank Net | 300 | 90.5 | 10.4 |
| Filterbank Net | 150 | 87.9 | 13.9 |

**Table 8.2: Filterbank Net vs FBCSP Accuracies.** Std is standard deviation over the 18 subjects used here. Results from Schirrmeister [76].

many artifacts. Furthermore, we evaluated 150Hz and 300 Hz as sampling rates here, in the remainder we will use 250 Hz.

The results in Table 8.2 show that the Filterbank net outperformed FBCSP by 2.4% (300 Hz) and 1.3% (150 Hz) respectively. Despite the good performance, we did not evaluate this network further after the master thesis due to the very large GPU memory requirement of our implementation and our interest in evaluating more expressive architectures not as tightly constrained to implement FBCSP steps.

### 8.5.3 ConvNets Reached FBCSP Accuracies

| DATASET | FREQUENCY RANGE (HZ) | FBCSP | DEEP | SHALLOW |
|---|---|---|---|---|
| BCIC IV 2a | 0–38 | 68.0 | +2.9 | +5.7* |
| BCIC IV 2a | 4–38 | 67.8 | +2.3 | +4.1 |
| HGD | 0–125 | 91.2 | +1.3 | -1.9 |
| HGD | 4–125 | 90.9 | +0.5 | +3.0* |
| Combined | 0–$f_{end}$ | 82.1 | +1.9* | +1.1 |
| Combined | 4–$f_{end}$ | 81.9 | +1.2 | +3.4** |

**Table 8.3: Deep and Shallow ConvNet vs. FBCSP Accuracies.** FBCSP decoding accuracies and difference of deep and shallow ConvNet accuracies to FBCSP results are given in percentage. BCIC IV 2a: BCI competition IV dataset 2a. HGD: High-Gamma Dataset. Frequency range is in Hertz. 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). Note that all P values below 0.01 in this study remain significant when controlled with false-discovery-rate correction at $\alpha = 0.05$ across all tests involving ConvNet accuracies.

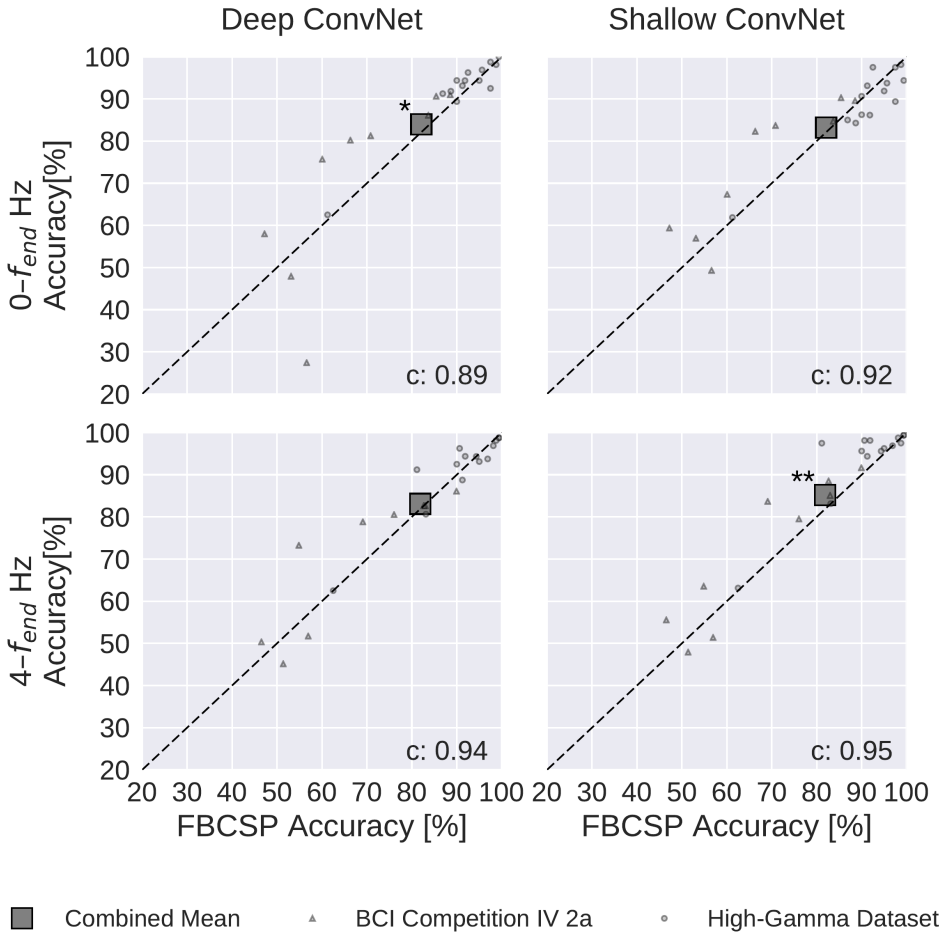



**Figure 8.1: FBCSP vs. ConvNet decoding accuracies.** Each small marker represents accuracy of one subject, the large square markers represent average accuracies across all subjects of both datasets. Markers above the dashed line indicate experiments where ConvNets performed better than FBCSP and opposite 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. Figure from Schirrmeister et al. [77].

Both the deep and the shallow ConvNets, with appropriate design choices (see Section 8.8), 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 $0 - f_{\text{end}}$ Hz and for the deep ConvNet on $4 - f_{\text{end}}$ Hz were not statistically significantly different

from FBCSP (see Figure 8.1). The deep ConvNet on $0 - f_{\text{end}}$ Hz and the shallow ConvNet on $4 - f_{\text{end}}$ 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. Note that all P values below 0.01 in this study remain significant when controlled with false-discovery-rate correction at $\alpha = 0.05$ across all tests involving ConvNet accuracies.

## 8.6 CONFUSION MATRICES ARE SIMILAR BETWEEN FBCSP AND 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 8.4: Decoding errors 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 errors 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 errors for all three decoding methods. In contrast, Hand (L and R) and Feet had a small number of errors 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 8.2). The majority of all mistakes were due to discriminating between Hand (L) / Hand (R) and Feet / Rest, see Table Table 8.4. 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 IV dataset 2a showed a larger variability and hence a less consistent pattern, possibly because of the much smaller number of trials.

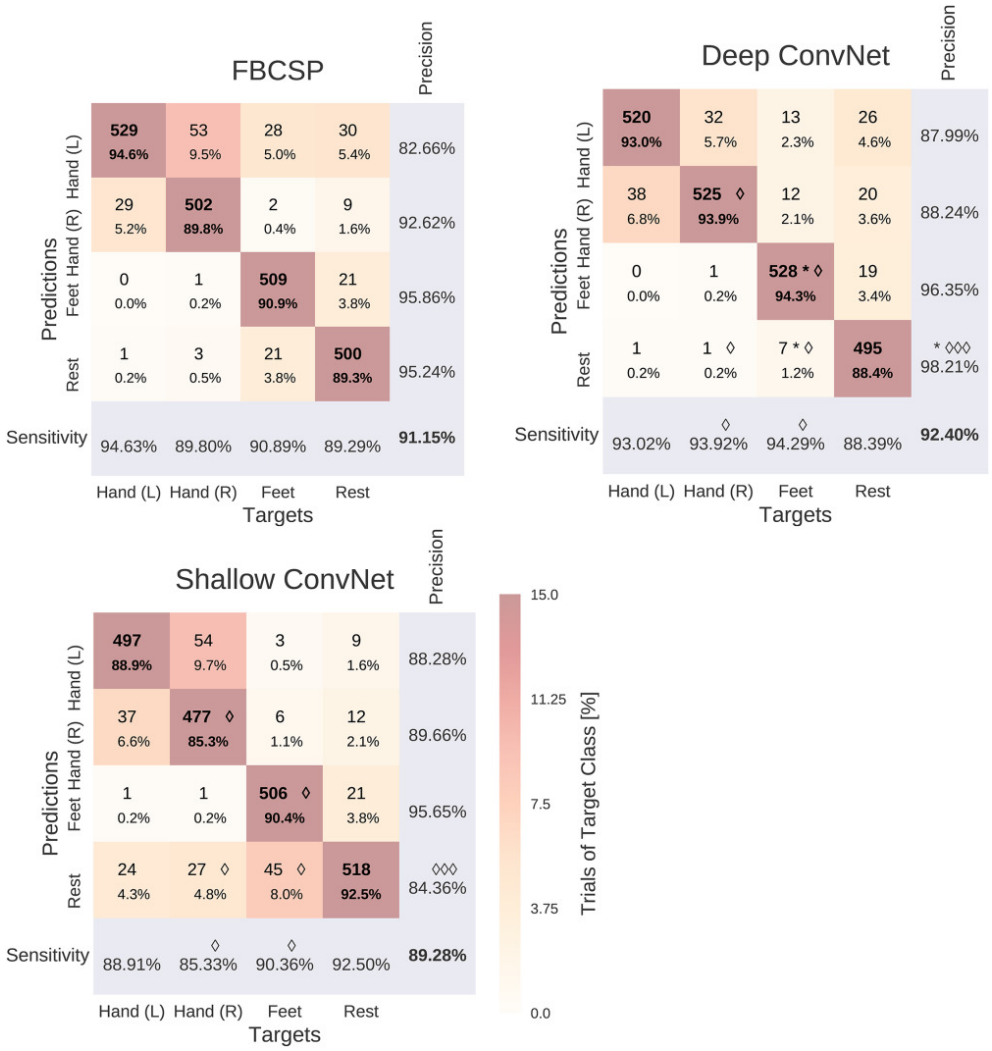



**Figure 8.2: Confusion matrices for FBCSP- and ConvNet-based decoding.** Results are shown for the High-Gamma Dataset, on $0^{\sim}f_{\text{end}}$ Hz. Each entry of row r and column c for upper-left 4×4-square: Number of trials of target r predicted as class c (also written in percent of all trials). Bold diagonal corresponds to correctly predicted trials of the different classes. Percentages and colors indicate fraction of trials of corresponding column (i.e., from trials of the corresponding target class). The lower-right value corresponds to overall accuracy. Bottom row corresponds to sensitivity : $\frac{\text{number of trials correctly predicted for class c}}{\text{number of trials for class c}}$. Rightmost column corresponds to precision: $\frac{\text{number of trials correctly predicted for class r}}{\text{number of trials predicted as class r}}$. Stars indicate statistically significantly different values between ConvNet decoding and FBCSP, diamonds between the shallow and deep ConvNets. P<0.05: ◇/*, P<0.01: ◇◇/**, P<0.001: ◇◇◇/***, Wilcoxon signed-rank test. Figure from Schirrmeister et al. [77].

## 8.7 RESIDUAL CONVNETS CAN BE COMPETITIVE WITH IMPROVED TRAINING

| DATASET | FREQUENCY RANGE (HZ) | FBCSP | RESNET BEFORE | RESNET NOW |
|---|---|---|---|---|
| BCIC IV 2a | 0–38 | 68.0 | -0.3 | +3.7 |
| BCIC IV 2a | 4–38 | 67.8 | -7.0* | -0.9 |
| HGD | 0–125 | 91.2 | -2.3* | +1.1 |
| HGD | 4–125 | 90.9 | -1.1 | 0.0 |
| Combined | 0–$f_{end}$ | 82.1 | -1.1 | +2.2 |
| Combined | 4–$f_{end}$ | 81.9 | -3.5* | -0.4 |

**Table 8.5: Residual ConvNet vs. FBCSP Accuracies.** Accuracies and significance stars have same meaning as before.

In our original study, residual ConvNets had underperformed our FBCSP baseline, in several settings even with statistical significance. However, later investigations revealed that better weight initializations, concretely scaling down the weight initialization more strongly, can lead to improved results. In work done for this thesis, I repeated the original experiments using our current decoding pipeline with the residual ConvNets. Table 8.5 reports the original accuracies and the accuracies obtained with our current codebase. Notably, residual ConvNets no longer substantially underperform FBCSP, even slightly outperforming it in the setting without highpass.

## 8.8 DESIGN CHOICES AFFECTED DECODING PERFORMANCE

Design choices substantially affected deep network accuracies on both datasets, meaning BCI Competition IV 2a and the High Gamma Dataset. Batch normalization and dropout significantly increased accuracies. This became especially clear when omitting both simultaneously Figure 8.3. 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

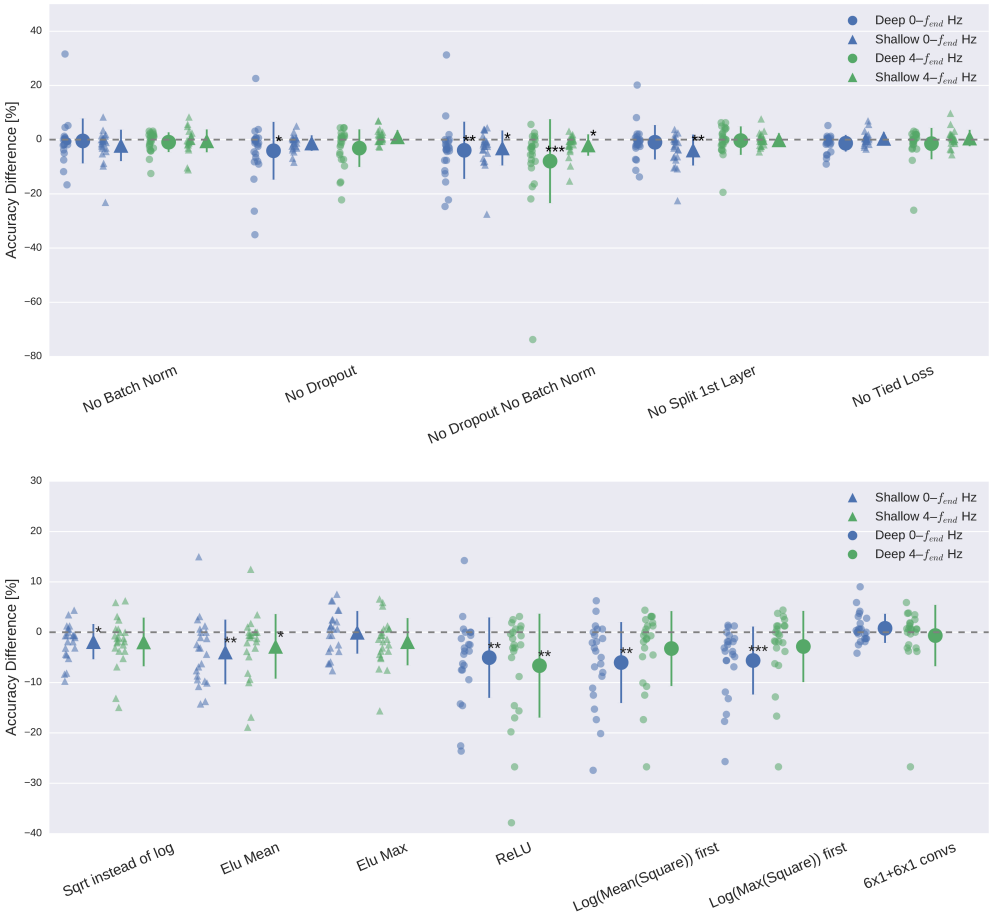



**Figure 8.3: Impact of ConvNet design choices on decoding accuracy.** Accuracy differences of baseline and design choices on x-axis for the $0 - f_{\text{end}}$ Hz and $4 - f_{\text{end}}$ Hz datasets. Each small marker represents accuracy difference for one subject, and each larger marker represents 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=). Top: Impact of design choices applicable to both ConvNets. All statistically significant differences were accuracy decreases from removing an aspect of our architecture. Notably, there was a clear negative effect of removing both dropout and batch normalization. Bottom: 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 from Schirrmeister et al. [77].

the strongest accuracy increase on the $0 - f_$end Hz data for the shallow ConvNet, where it is also the only statistically significant difference ($p < 0.01$).

## 8.9 CROPPED TRAINING STRATEGY IMPROVED DEEP CONVNET ON HIGHER FREQUENCIES

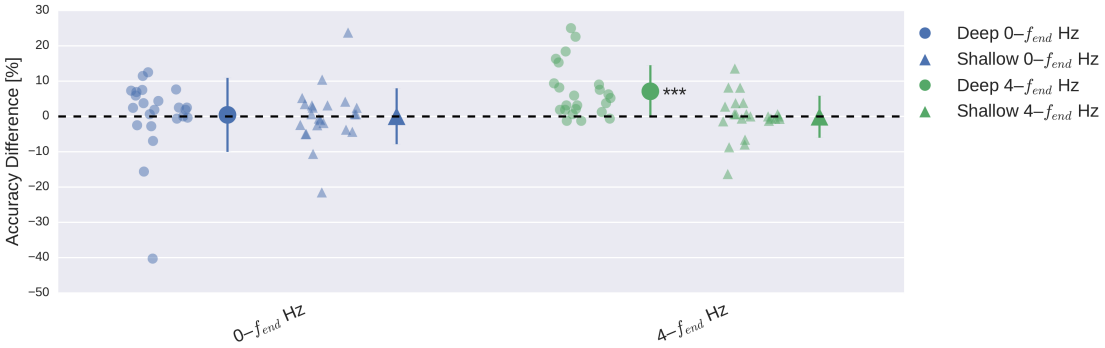



**Figure 8.4: Impact of training strategy (cropped vs trial-wise training) on accuracy.** Accuracy difference for both frequency ranges and both ConvNets when using cropped training instead of trial-wise training. Other conventions as in **??**. Cropped training led to better accuracies for almost all subjects for the deep ConvNet on the $4 - f_{\text{end}}$-Hz frequency range. Figure from Schirrmeister et al. [77].

Cropped training increased accuracies statistically significantly for the deep ConvNet on the $4 - f_{\text{end}}$-Hz data (p<1e-5, Wilcoxon signed-rank test, see Figure 8.4). In all other settings ($0 - f_{\text{end}}$-Hz data, shallow ConvNet), the accuracy differences were not statistically significant (p>0.1) and showed a lot of variation between subjects.

## 8.10 RESULTS ON BCI COMPETITION IV 2B

| FBCSP | DEEP CONVNET | SHALLOW CONVNET |
|---|---|---|
| 0.599 | -0.001 | +0.030 |

**Table 8.6: Kappa values on the BCIC IV 2b dataset.** ConvNet kappa values show the difference to the FBCSP kappa value.

To ensure that the results also generalize to further datasets and also rule out hyperparameter overfitting, the FBCSP pipeline and the deep network pipelines

were applied with the exact same hyperparameters on BCI Competition IV 2b. A few choices like the use of the decoding time window had been done after already seeing results from the evaluation sets of the High-Gamma dataset and the BCIC IV 2a dataset, hence it was valuable to validate the results on the BCIC IV 2b dataset. Results in Table 8.6 show that the networks perform as good or better than FBCSP. Results on further datasets, also non-movement-decoding datasets are presented in the next chapter Chapter 9.

## 8.11 CONVNET-INDEPENDENT VISUALIZATIONS

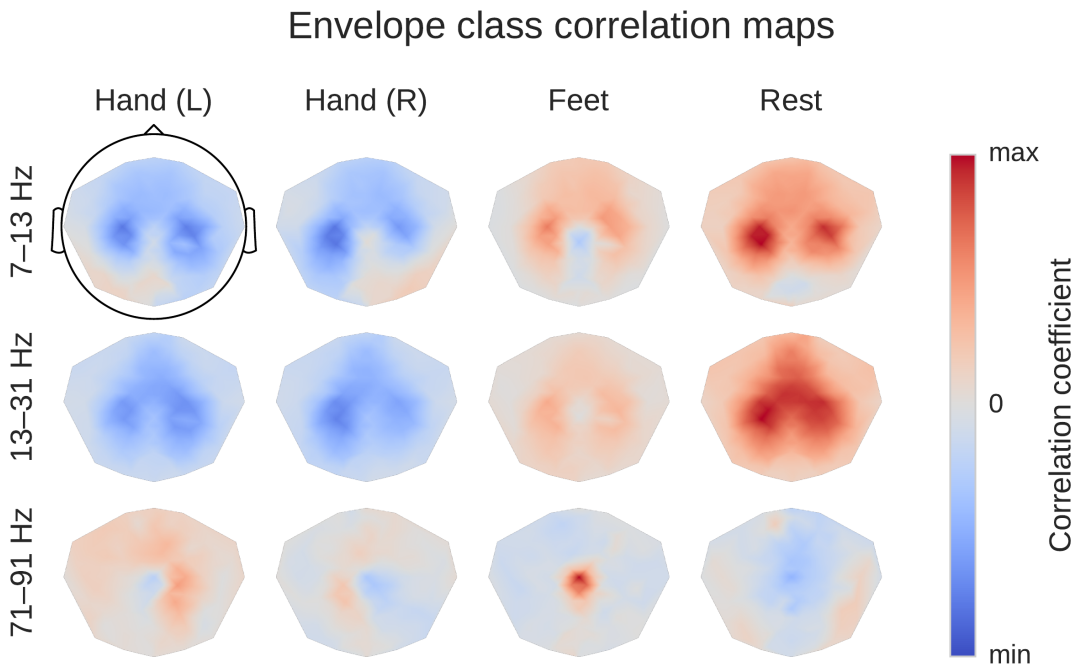



**Figure 8.5: Average envelope correlations over subjects from the high-gamma dataset.** Colormaps are scaled per frequency band/row. This is a ConvNet-independent visualization. Scalp plots show spatial distributions of class-related spectral amplitude changes well in line with the literature. Figure from Schirrmeister et al. [77].

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. For that, we first computed the moving average of the squared envelope in narrow frequency bands via the Hilbert transform as a measure of the power in those frequency bands. Then we computed linear correlations of these

moving averages with the class label. This results in frequency-resolved envelope-class label correlations.

We found the expected overall scalp topographies (see Figure 8.5) to show physiologically plausible patterns. 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 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).

## 8.12 AMPLITUDE PERTURBATION VISUALIZATIONS

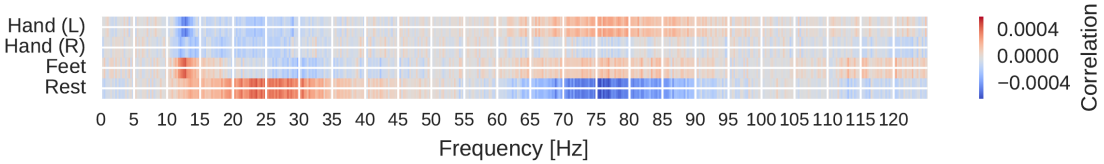



**Figure 8.6: Input-perturbation network-prediction correlations for all frequencies for the deep ConvNet, per class.** Plausible correlations, for example, rest positively, other classes negatively correlated with the amplitude changes in frequency range from 20 to 30 Hz. Figure from Schirrmeister et al. [77].

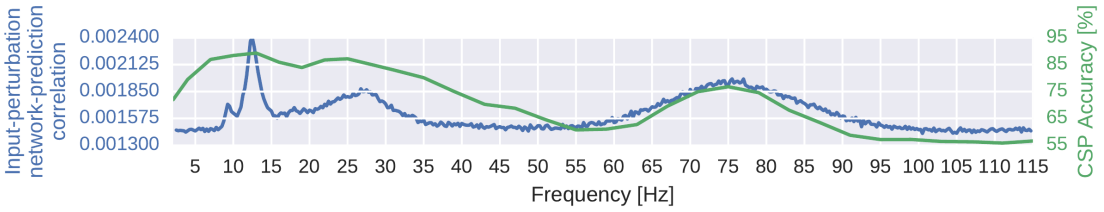



**Figure 8.7: Absolute input-perturbation network-prediction correlation frequency profile for the deep ConvNet.** Mean absolute correlation value across classes. CSP binary decoding accuracies for different frequency bands for comparison, averaged across subjects and class pairs. Peaks in alpha, beta, and gamma band for input-perturbation network-prediction correlations and CSP accuracies. Figure from Schirrmeister et al. [77].

Our amplitude perturbation visualizations show that the network have learned to extract commonly used spectral amplitude features.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 in Figure 8.6.

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 (Figure 8.7). Similar peaks were seen in the means of the CSP binary decoding accuracies for the same frequency range.

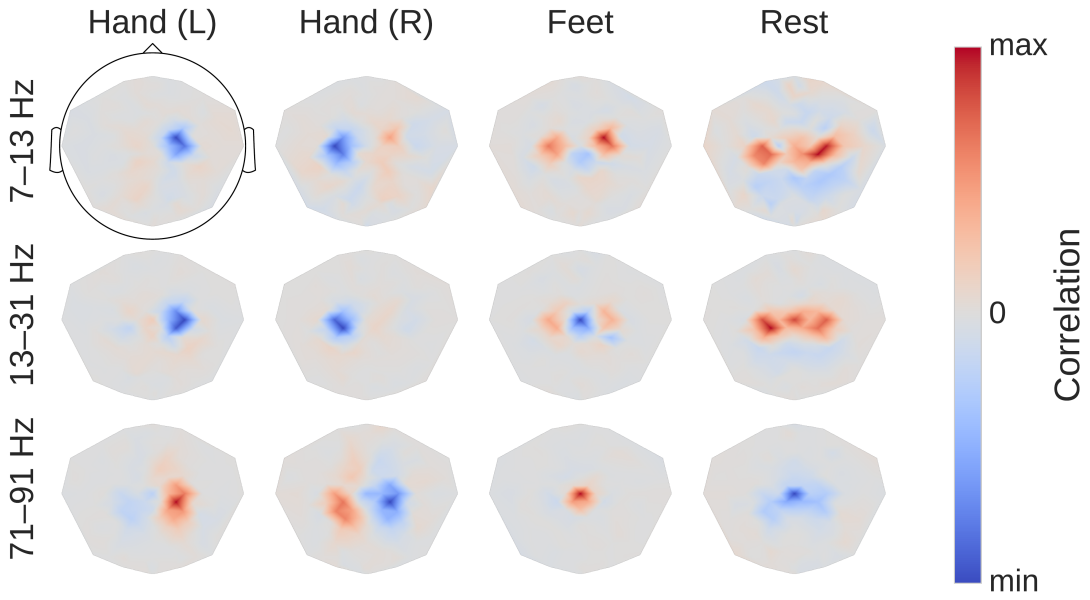



**Figure 8.8: Input-perturbation network-prediction correlation maps for the deep ConvNet.** Correlation of class predictions and amplitude changes. Averaged over all subjects of the High-Gamma Dataset. Colormaps are scaled per scalp plot. Plausible scalp maps for all frequency bands, for example, contralateral positive correlations for the hand classes in the gamma band. Figure from Schirrmeister et al. [77].

Third, scalp maps of the input-perturbation effects on network predictions for the different frequency bands, as shown in Figure 8.8, show spatial distributions expected for motor tasks in the alpha, beta and — for the first time for such a noninvasive EEG decoding visualization — for the high gamma band. These scalp maps directly reflect the behavior of the ConvNets and one needs to 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 Section 6.4 and [31]. Nevertheless, for the first time in noninvasive EEG, these maps clearly revealed the global somatotopic organization of causal contributions of motor cortical gamma band activity to decoding right and left hand and foot movements. Interestingly, these maps revealed highly focalized patterns, particularly during hand movement in the gamma frequency range (Figure 8.8, first plots in last row), in contrast to the more diffuse patterns in the conventional task-related spectral analysis as shown in Figure 8.5.

## 8.13 INTERNAL REPRESENTATIONS OF AMPLITUDE AND PHASE

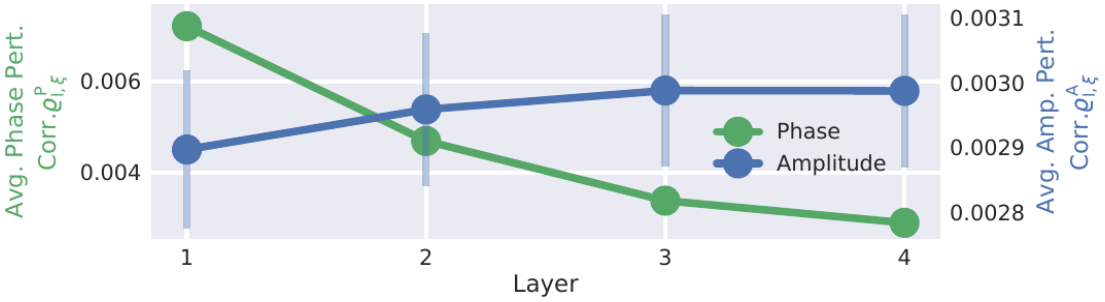



**Figure 8.9: Mean phase and amplitude perturbation correlations over layers.** Curves show mean perturbation correlation over all frequencies for each layer. Scales are different and written on the left and right y-axes. The error bars show the standard error over the subjects. Standard errors for phase and amplitude are similar, but much higher relatively in the amplitude correlation scale and therefore only visible there. In their respective scale, curves show a clearly inverse behavior over the layers with increasing amplitude correlations and decreasing phase correlations. Figure from Hartmann, Schirrmeister, and Ball [30].

The perturbation analysis showed that the earlier layers represent more phase-specific features than later layers, while the later layers represent more phase-invariant amplitude features than the early layers. Figure 8.9 shows the average absolute phase perturbation correlation and amplitude perturbation correlation over the 4 convolutional layers. The figure shows a clearly opposing development of their respective average values across layers, with increasing amplitude perturbation correlations and decreasing phase perturbation correlations.

The perturbation correlations for individual frequencies showed a strong phase perturbation correlation in the earlier layers 1 and 2 to phases in the alpha, beta, and high gamma range (see Figure 8.10 ). The overall phase perturbation correlation was highest in layer 1 and gradually became lower over layers 2, 3, and 4. Interestingly, for each frequency band (alpha, beta, and high gamma), there was one specific layer in

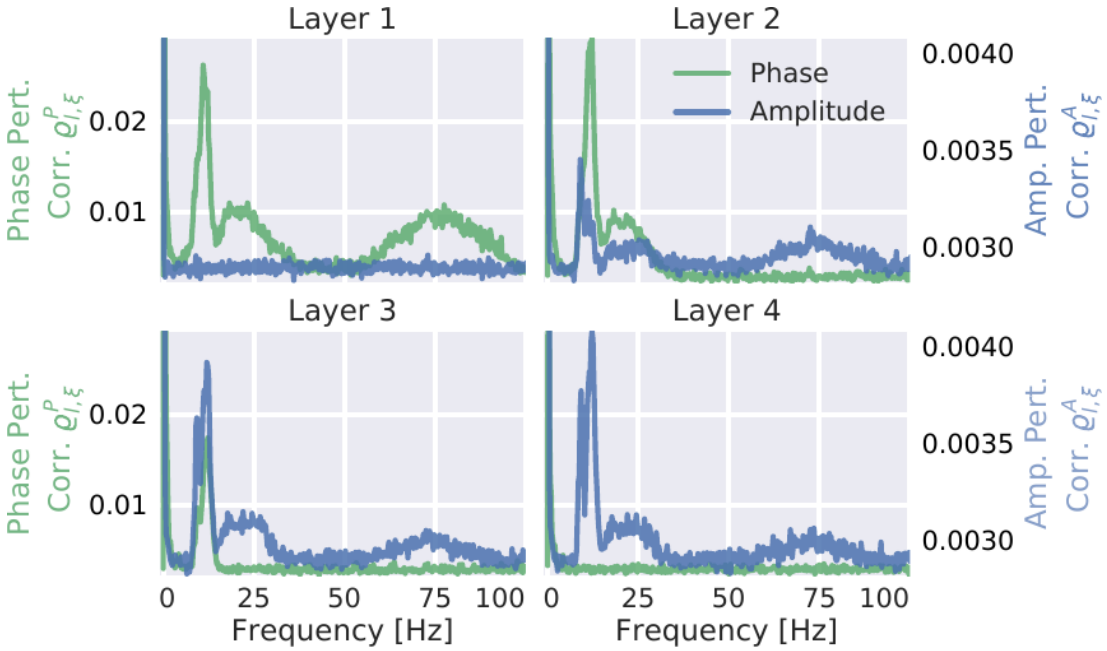



**Figure 8.10: Mean of absolute phase and amplitude perturbation correlations for individual frequencies.** The two correlation types have different scales, denoted by the left and right y-axis. As in Figure 8.9, a clearly inverse relation between amplitude and phase correlations is visible. This visualization additionally showed that alpha (7-13 Hz), beta (13-30 Hz), and high gamma (50-100 Hz) frequency ranges each have a specific layer in which their phase correlation vanishes and their amplitude correlation saturates (high gamma in layer 2, beta in layer 3, and alpha in layer 4). Figure from Hartmann, Schirrmeister, and Ball [30].

which the phase perturbation correlations vanished completely. Phase perturbation correlations for high gamma vanished in layer 2, correlations for beta vanished in layer 3, and correlations for alpha vanished in layer 4. This vanishing of phase correlations for high gamma in layer 2 could be observed in all subjects. For 4 subjects, alpha did already vanish together with beta in layer 3.

An opposite behavior could be observed for amplitude correlations. Notable phase insensitive amplitude perturbation correlations are only emerging in layer 2. Amplitude correlations of individual frequency bands peaked and saturated in the same layers in which the phase correlation vanished: High gamma amplitude perturbation correlation saturated in layer 2, beta in layer 3, and alpha in layer 4.

A potential underlying reason may be the use of max pooling with stride 3 after layers 1,2 and 3. The resulting temporal frequency resolutions of the intermediate representations when only taking into account the pooling strides are $\frac{250\text{Hz}}{3} \approx 83\text{Hz}$, $\frac{250\text{Hz}}{3^2} \approx 28\text{Hz}$ and $\frac{250\text{Hz}}{3^3} \approx 9\text{Hz}$. The corresponding Nyquist frequencies of approximately 41.5Hz, 14Hz and 4.5Hz seem to correspond to frequency cutoffs above which

the network is no longer phase-sensitive. However, note that the network is in principle capable of retaining phase sensitivity even in the presence of max pooling by shifting temporal information to its channel dimension.

## 8.14 MAXIMALLY ACTIVATING UNITS

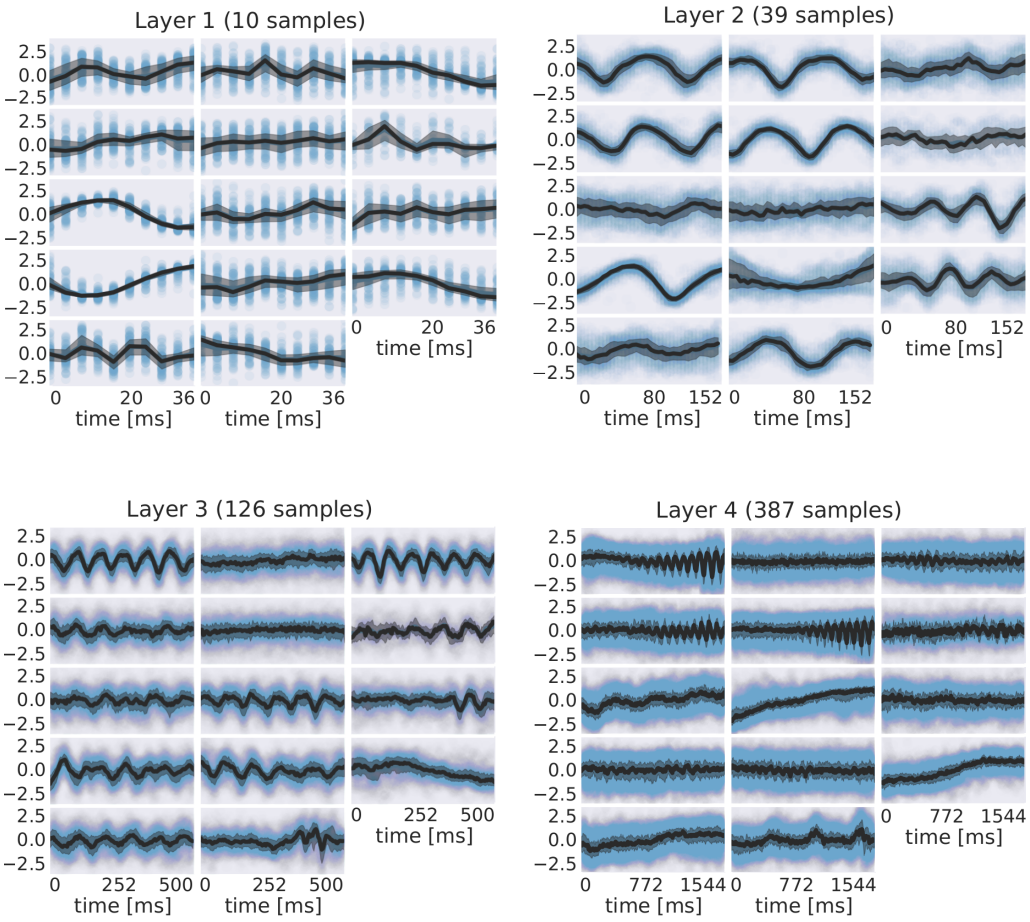



**Figure 8.11: EEG signals in most-activating input windows.** Taken from one randomly sampled filter in each layer for each subject. Blue points are the standard scores of all most-activating input windows for a filter. Their median is shown in black and the interquartile range as a gray shaded area. Medians in earlier layers often resemble parts of or complete sinusoids while medians in later layers resemble more complex patterns. Figure from Hartmann, Schirrmeister, and Ball [30].

In addition to examining how the individual layers respond to frequency-specific phase and amplitude, we were also interested in other characteristic features that might be learned by filters. To investigate other features than frequency-specific phase or amplitude of sinusoidal signals, we visually inspected the most-activating input windows of a filter and their median values for each timepoint. Figure 8.11 shows the most-activating input windows of one randomly sampled filter for each subject and layer. We show each such set of most activating input windows here at a representative electrode. For several filters, a clearly defined structure was present in the median. The median plots for layer 1 show several examples of sinusoidal shapes in different frequencies. Medians of higher frequencies were sharper and the variance across input windows is larger. Medians of layer 2 revealed several smooth alpha sinusoids, but also examples of beta waves. In addition to that, there were some flat medians without an easily interpretable periodicity or other temporal structure. In layer 3, there were mostly examples for alpha waves. For some medians in layer 4, more complex temporal patterns emerged. Those medians were relatively flat at the beginning of the input windows, but showed an oscillatory pattern with increasing amplitude in the later parts of the input windows. Also, the oscillatory patterns in some examples resembled mu waves more closely than pure sinusoids. Such patterns were found in several subjects.

## 8.15 BRAINDECODE

Work performed for this study also later led to the creation of the open-source EEG deep learning library Braindecode, now with contributions from multiple research groups and available at `https://github.com/braindecode/braindecode/`.

**Open Questions**

- How well do ConvNets perform on other decoding tasks?
- Do they work on tasks where oscillatory features are less important?
- Do they work on non-trial based tasks like decoding pathology?

# 9 GENERALIZATION TO OTHER TASKS

Our architectures generalize well to a wide variety of decoding tasks

- Perform similar or better than common feature-based algorithms on mental imageries, error decoding, auditory evoked potentials
- Also perform well on intracranial EEG
- Deep networks performs a bit better than shallow network on average across tasks
- EEGNet architecture developed by others also performs well
- Networks can be used in an online BCI scenario

After our initial work designing and evaluating convolutional neural networks for movement decoding from EEG, we evaluated the resulting networks on a wide variety of other EEG decoding tasks found that they generalize well to a large number of settings such as error-related decoding, online BCI control or auditory evoked potentials and also work on intracranial EEG.

Text and content condensed from a number of publications, namely Schirrmeister et al. [77], Völker et al. [96], Burget et al. [13], Volker et al. [95], Behncke et al. [8], Wang et al. [98] and Heilmeyer et al. [33]. In all of these works except Schirrmeister et al. [77], I was not the main contributor, I assisted in adapting the code and training for the various settings and helped in the writing process.

## 9.1 DECODING DIFFERENT MENTAL IMAGERIES

| FBCSP | DEEP CONVNET | SHALLOW CONVNET |
|---|---|---|
| 71.2 | +1.0 | -3.5 |

**Table 9.1: Accuracies on the Mixed-Imagery dataset.** ConvNet accuracies show the difference to the FBCSP accuracy. Results from Schirrmeister et al. [77].

The Mixed Imagery Dataset (MID) was obtained from 4 healthy subjects (3 female, all right-handed, age 26.75±5.9 (mean±std)) with a varying number of trials (S1: 675, S2: 2172, S3: 698, S4: 464) of imagined movements (right hand and feet), mental

rotation and mental word generation. All details were the same as for the High Gamma Dataset, except: a 64-electrode subset of electrodes was used for recording, recordings were not performed in the electromagnetically shielded cabin, thus possibly better approximating conditions of real-world BCI usage, and trials varied in duration between 1 to 7 seconds. The dataset was analyzed by cutting out time windows of 2 seconds with 1.5 second overlap from all trials longer than 2 seconds (S1: 6074 windows, S2: 21339, S3: 6197, S4: 4220), and both methods were evaluated using the accuracy of the predictions for all the 2-second windows for the last two runs of roughly 130 trials (S1: 129, S2: 160, S3: 124, S4: 123).

For the mixed imagery dataset, we find the deep ConvNet to perform slightly better and the shallow ConvNet to perform slightly worse than the FBCSP algorithm, as can be seen in Table 9.1.

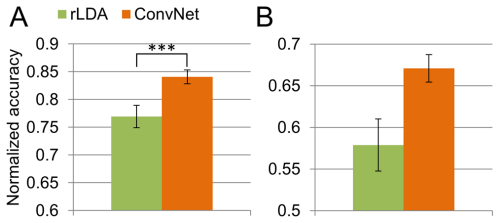
## 9.2 DECODING ERROR-RELATED SIGNALS

### 9.2.1 Decoding Observation of Robots Making Errors

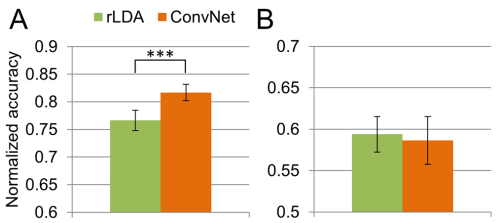
| ROBOT TASK | TIME INTERVAL | DEEP CONVNET | RLDA | FBCSP |
|---|---|---|---|---|
| Pouring Liquid | 2-5s | 78.2 ± 8.4 | 67.5 ± 8.5 | 60.1 ± 3.7 |
| Pouring Liquid | 3.3-7.5s | 71.9 ± 7.6 | 63.0 ± 9.3 | 66.5 ± 5.7 |
| Lifting Ball | 4.8-6.3s | 59.6 ± 6.4 | 58.1 ± 6.6 | 52.4 ± 2.8 |
| Lifting Ball | 4-7s | 64.6 ± 6.1 | 58.5 ± 8.2 | 53.1 ± 2.5 |

**Table 9.2: Accuracies for robot error observation.** Task was to decode whether a person watches a successful or unsuccessful robot-liquid pouring or ball-lifting. Results from Behncke et al. [7].

In this study, we aimed to classify whether a person had watched a video of a successful or an unsuccessful attempt of a robot performing one of two tasks (lifting a ball or pouring liquid) based on EEG recorded during the video observation. We compared the performance of our deep ConvNet to that of regularized linear discriminant analysis (rLDA) and FBCSP on this task. Our results, presented in Table 9.2, demonstrate that the deep ConvNet outperformed the other methods for both tasks and both decoding intervals.




**(a) Comparison of within-subject decoding by rLDA and deep ConvNets.** Error bars show the SEM. A) Eriksen flanker task (mean of 31 subjects), last 20% of subject data as test set. Deep ConvNets were 7.12% better than rLDA, pval = 6.24 *10-20 (paired t-test). B) Online GUI control (mean of 4 subjects), last session of each subject as test data. Figure from [96].




**(b) Mean normalized decoding accuracy on unknown subjects.** Error bars show the SEM. A) Eriksen flanker task, trained on 30 subjects, tested on 1 subject. Deep ConvNets were 5.05% better than rLDA, p = 3.16 *10-4 (paired t-test). B) Online GUI control. Trained on 3 subjects, tested on the respective remaining subject. Figure from cite:t'volker2018deep'.

**Figure 9.1: Error decoding accuracy on Eriksen flanker task and online GUI control.**

### 9.2.2 Decoding of Eriksen Flanker Task Errors and Errors during Online GUI Control

In two additional error-related decoding experiments, we evaluated an Eriksen flanker task and errors during an the online control of a graphical user interface through a brain-computer-interface. In the Eriksen flanker task, the subjects were asked to press the left or right button on a gamepad depending on whether an 'L' or an 'R' was the middle character of a 5-letter string displayed on the screen. For the online graphical user interface (GUI) control, the subjects were given an aim to reach using the GUI, also see **??**. They had to think of one of the classes of the aforementioned Mixed Imagery Dataset to choose one of four possible GUI actions. The correct GUI action was always determined by the specificed aim given to the subject, hence an erroneous action could be detected. The decoding task in this paper was to distinguish whether the BCI-selected action was correct or erroneous. Results in Figure 9.1a and Figure 9.1b show that deep ConvNets outperform rLDA in all settings except cross-subject error-decoding for online GUI control, where the low number of subjects (4) may prevent the ConvNets to learn enough to outperform rLDA.

## 9.3 PROOF-OF-CONCEPT ASSISTIVE SYSTEM

We also evaluated the use of our deep ConvNet as part of an assistive robot system where the brain-computer interface was sending high-level commands to a robotic arm. In this proof-of-concept system, the robotic arm could be instructed by the user

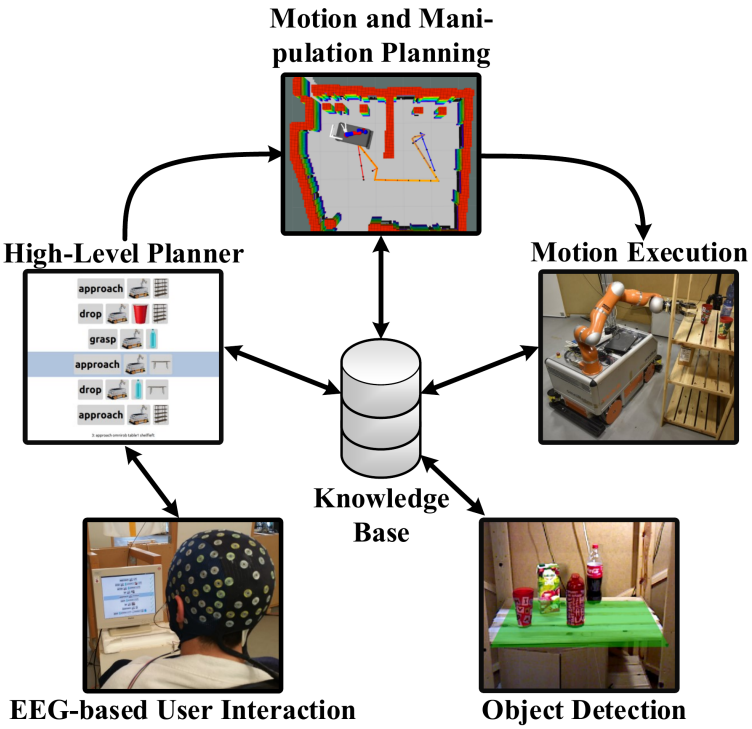



Figure 9.2: **Overview of the proof-of-concept assistive system. The system uses the deep ConvNet in the BCI component.** Robotic arm could be given high-level commands via the BCI, high-level commands were extracted from a knowledge base. The commands were then autonomously planned and executed by the robotic arm. Figure from Burget et al. [13]

via the BCI to fetch a cup and directly move the cup to the persons mouth to drink from it. An overview can be seen in Figure 9.2. Results from Table 9.4 show that 3 out of 4 subjects had a command accuracy of more than 75% and were able to reach the target using less than twice the steps of the minimal path through the GUI (path optimality > 50%).

## 9.4 INTRACRANIAL EEG DECODING

### 9.4.1 Intracranial EEG Decoding of Eriksen Flanker Task

We further evaluated whether the same networks developed for noninvasive EEG decoding can successfully learn to decode intracranial EEG. Therefore, in one work we used the same Eriksen flanker task as described in Section 9.2.2, but recorded intracranial EEG from 23 patients who had pharmacoresistant epilepsy [95]. Results

| SUBJECT | RUNS | ACCU-RACY [%] | TIME [S] | STEPS | PATH OPTI-MALITY [%] | TIME / STEP [S] |
|---|---|---|---|---|---|---|
| S1 | 18 | 84.1 ± 6.1 | 125 ± 84 | 13.0 ± 7.8 | 70.1 ± 22.3 | 9 ± 2 |
| S2 | 14 | 76.8 ± 14.1 | 150 ± 32 | 10.1 ± 2.8 | 91.3 ± 12.0 | 9 ± 3 |
| S3 | 17 | 82.0 ± 7.4 | 200 ± 159 | 17.6 ± 11.4 | 65.7 ± 28.9 | 11 ± 4 |
| S4 | 3 | 63.8 ± 15.6 | 176 ± 102 | 26.3 ± 11.2 | 34.5 ± 1.2 | 6 ± 2 |
| Average | 13 | 76.7 ± 9.1 | 148 ± 50 | 16.7 ± 7.1 | 65.4 ± 23.4 | 9 ± 2 |

**Table 9.3: Results for BCI control of the GUI.** Accuracy is fraction of correct commands, time is time per command, steps is steps needed to reach the aim, path optimality is ratio of miniminally needed nubmer of steps to actually used number of steps when every step is optimal, and time/step is time per step. Results from Burget et al. [13].

| CLASSIFIER | BALANCED ACCURACY | ACCURACY CORRECT CLASS | ACCURACY ERROR CLASS |
|---|---|---|---|
| Deep4Net | 59.28 ± 0.50 | 69.37 ± 0.44 | 49.19 ± 0.56 |
| ShallowNet | 58.42 ± 0.32 | 74.83 ± 0.25 | 42.01 ± 0.40 |
| EEGNet | 57.73 ± 0.52 | 57.78 ± 0.48 | 57.68 ± 0.56 |
| rLDA | 53.76 ± 0.32 | 76.12 ± 0.26 | 31.40 ± 0.38 |
| ResNet | 52.45 ± 0.21 | 95.47 ± 0.14 | 09.43 ± 0.28 |

**Table 9.4: Results for single-channel intracranial decoding of errors during an Eriksen flanker task.** Balanced Accuracy is the mean of the accuracies for correct class ground truth labels and error class ground truth labels. Results from **intracranial-error-results-table**.

for single-channel decoding **??** show the deep and shallow ConvNet to clearly outperform rLDA (59.3%/58.4% vs. 53.8%) , whereas the residual ConvNet has low accuracy (52.5%). In contrast, results for all-channel decoding Figure 9.3 show the residual ConvNet to perform well with the residual ConvNet and the deep ConvNet outperforming the shallow ConvNet (72.1% and 73.7% vs. 60.3% class-normalized accuracies (average over per-class accuracies)).

### 9.4.2 Transfer Learning for Intracranial Error Decoding

We further tested the potential of ConvNets to transfer knowledge learned from decoding intracranial signals in error-decoding paradigm to decoding signals in another a different error-decoding paradigm [8]. The two error-decoding paradigms were the aforementioned Eriksen flanker task (EFT) and a car driving task (CDT), where subjects had to use a steering wheel to steer a car in a computer game and

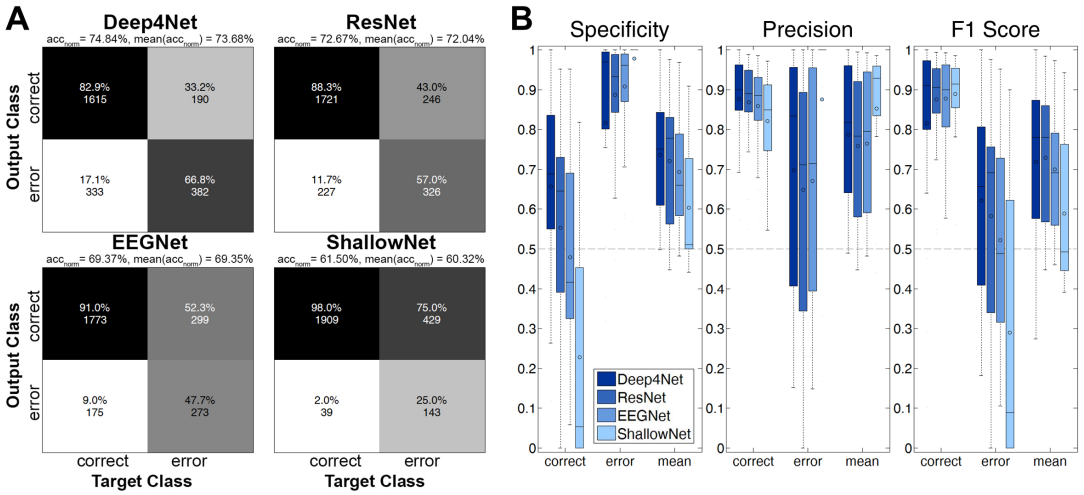



**Figure 9.3: Results for all-channel intracranial decoding of errors during an Eriksen flanker task.** Here, the classifiers were trained on all available channels per patient. A) Confusion matrices of the four models used for decoding. The matrices display the sum of all trials over the 24 recordings. On top of the matrices, the class-normalized accuracy (average over per-class accuracies) over all trials, i.e., $\text{acc}_{\text{norm}}$, and the mean of the single recordings' normalized accuracy, i.e., $\text{mean}(\text{acc}_{\text{norm}})$ is displayed; please note that these two measures differ slightly, as the patients had a varying number of total trials and trials per class. B) Box plots for specificity, precision and F1 score. The box represents the interquartile range (IQR) of the data, the circle within the mean, the horizontal line depicts the median. The lower whiskers include all data points that have the minimal value of $25^{\text{th}}\text{percentile} - 1.5 \cdot \text{IQR}$, the upper whiskers include all points that are maximally $75^{\text{th}}\text{percentile} + 1.5 \cdot \text{IQR}$. Figure from Volker et al. [95].

avoid hitting obstacles, where hitting an obstacle was considered an error event (see Figure 9.4). Results in Figure 9.5 show that pretraining on CDT helps EFT decoding when few EDT data is available.

### 9.4.3 Microelectrocorticography Decoding of Auditory Evoked Responses in Sheep

In this study, we evaluated the ConvNets for decoding auditory evoked responses played to a sheep that was chronically implanted with a

$\mu$ECoG-based neural interfacing device [98].

3-seconds-long sounds were presented to the sheep and two decoding tasks were defined from those 3 seconds as well as the second immediately before and after the playing of the sound. The first decoding task was to distinguish the 3 seconds when the sound was playing from the second immediately before and the second

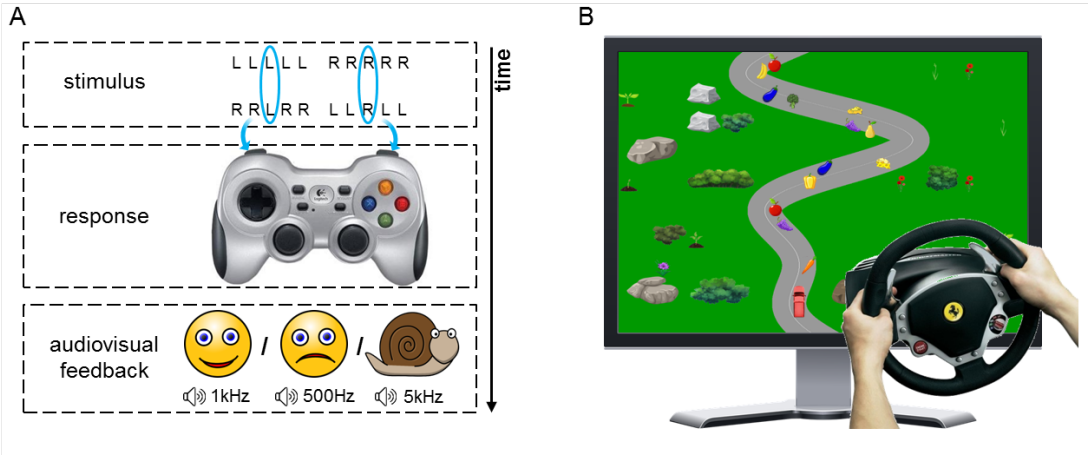



**Figure 9.4: Sketch of the Eriksen flanker task (A) and screenshot of the car driving task (B).** Figure from Behncke et al. [8].

immediately after the sound. The second task was distinguishing the first, second and third second of the playing of the sound to discriminate early, intermediate and late auditory evoked response (see Figure 9.6). Results in Figure 9.7 show that the deep ConvNet can perform as good as FBSCP and rLDA, and perform well on both tasks, whereas rLDA performs competitively only on the first and FBSCP only on the second task.

## 9.5 EVALUATION ON LARGE-SCALE TASK-DIVERSE DATASET

We also compared the deep and shallow ConvNet architectures as well as EEGNet on six classification tasks with more than 90000 trials in total (see Table 9.5) [33]. The datasets tasks were all recorded in our lab and included the high-gamma dataset, three error-related tasks described before (Eriksen flanker task, robot grasping and robot pouring observations) as well as two tasks on semantic processing. In the semantic processing dataset, the classification tasks were to distinguish different types of words that a subject silently repeated [66]. The first task was to distinguish existing real words from nonexisting pseudowords. The second classification task was to distingiush three semantic categories (food, animals, tools) the word may belong to. The evaluation code for all models always used the original code and hyperparameters from the original studies in order to ensure a fair comparison. Results show that the deep ConvNet and the more recent version of EEGNet (EEGNetv2) perform similarly well, with shallow and an older version of EEGNet performing slightly worse, see Figure 9.8, Figure 9.9 and Table 9.6.

| NAME (ACRONYM) | #CLASSES | TASK TYPE | #SUBJECTS | TRIALS PER SUBJECT | CLASS BALANCE |
|---|---|---|---|---|---|
| High-Gamma Dataset (Motor) | 4 | Motor task | 20 | 1000 | balanced |
| KUKA Pouring Observation (KPO) | 2 | Error observation | 5 | 720-800 | balanced |
| Robot-Grasping Observation (RGO) | 2 | Error observation | 12 | 720-800 | balanced |
| Error-Related Negativity (ERN) | 2 | Eriksen flanker task | 31 | 1000 | 1/2 up to 1/15 |
| Semantic Categories | 3 | Speech imagery | 16 | 750 | balanced |
| Real vs. Pseudo Words | 2 | Speech imagery | 16 | 1000 | 3/1 |

**Table 9.5: Datasets for the large-scale evaluation framework.** Used in Heilmeyer et al. [33].

| NETWORK | MEAN ACCURACY | MEAN NORMALIZED ACCURACY |
|---|---|---|
| Deep ConvNet | 70.08% ± 20.92% | 1.00 ± 0.05 |
| EEGNetv2 | 70.00% ±18.86% | 1.02 ± 0.08 |
| EEGNet | 67.71% ± 19.04% | 0.98 ± 0.06 |
| Shallow ConvNet | 67.71% ±19.04% | 0.99 ± 0.06 |

**Table 9.6: Dataset-averaged results for the large-scale evaluation of deep ConvNet, shallow ConvNet and two versions of EEGNet.** Accuracies are normalized to the average of the accuracies of all models. Results from Heilmeyer et al. [33].

Open Questions

- How do these networks perform on non-trial-based tasks like pathology decoding?

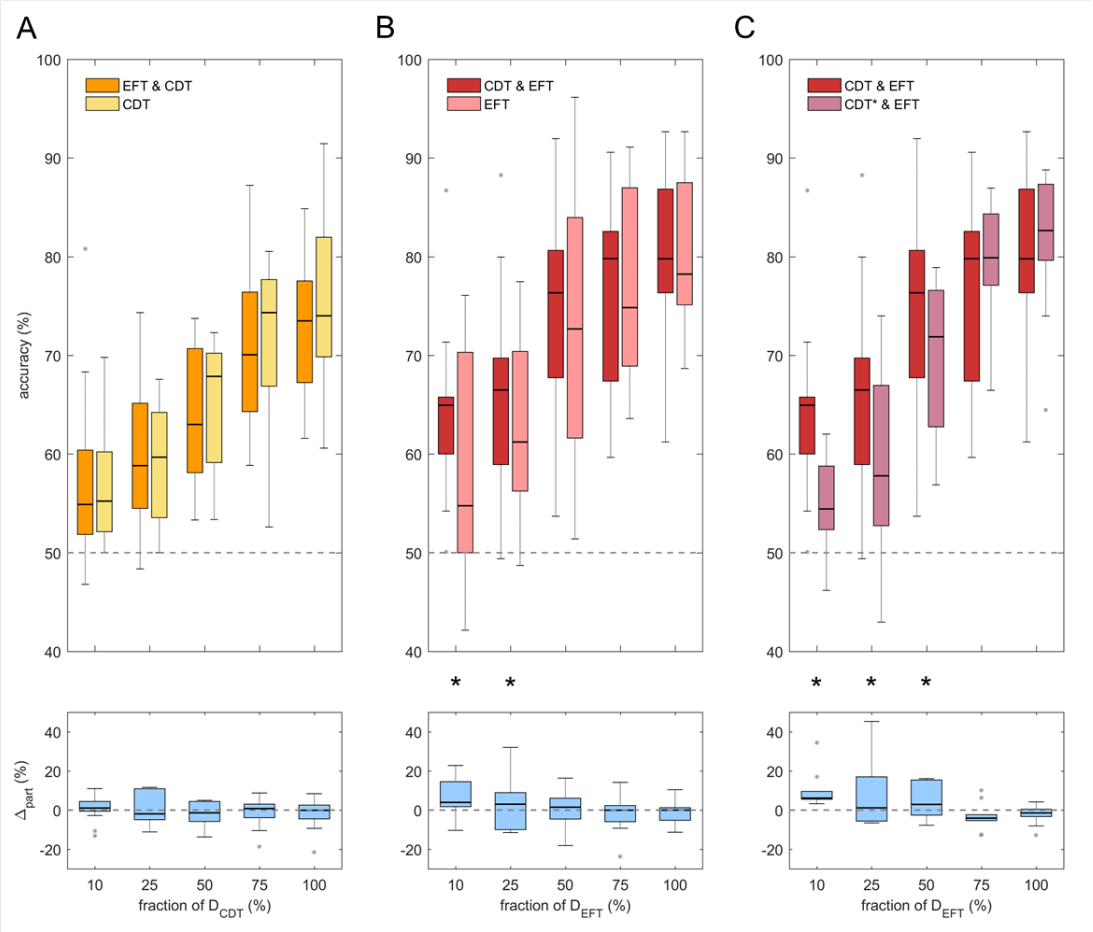



Figure 9.5:

**Results for transfer learning on the Eriksen flanker task (EFT) and the car driving task (CDT).** All results are computed for a varying fraction of available data for the target decoding task (bottom row). **A** compares CDT accuracies after training only on CDT or pretraining on EFT and finetuning on CDT. **B** compares EFT accuracies after only training on EFT or after pretraining on CDT and finetuning on EFT. As a sanity check for the results in **B**, **C** compares EFT accuracies when pretraining on original CDT data and finetuning on EFT to pretraining on CDT data with shuffled labels (CDT*) and finetuning on EFT. Results show that pretraining on CDT helps EFT decoding when little EFT data is available. Figure from Behncke et al. [8].

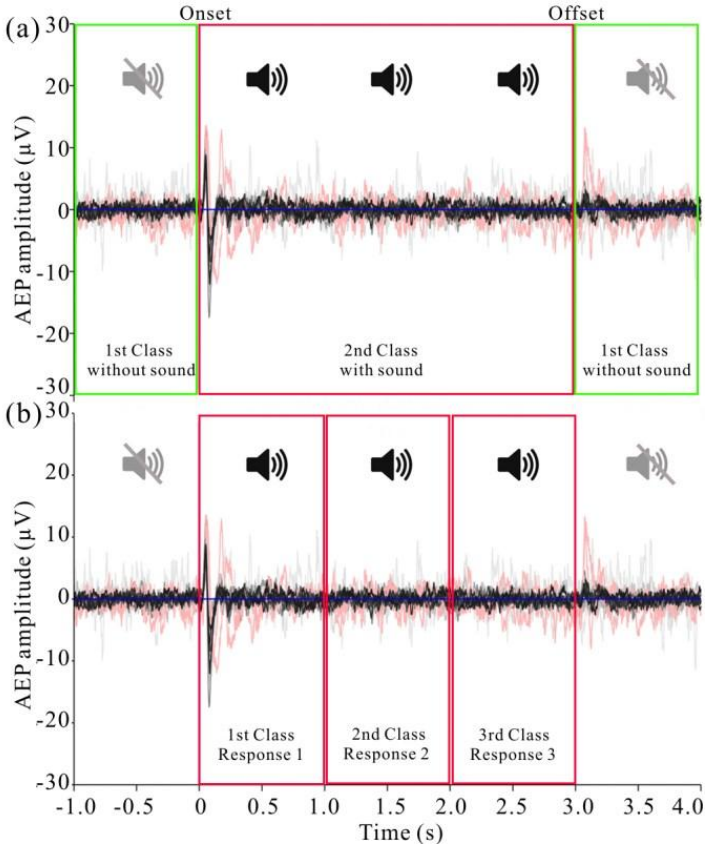



**Figure 9.6: Overview over decoding tasks for auditory evoked responses in a sheep.** First task (top) was to distingish 3 seconds when the sound was playing from the second before and the second after. Second task (bottom) was to distinguish the first, second and third second during theplaying of the sound. Signals are averaged responses from one electrode during different days, with black and grey being signals while the sheep was awake and red ones while the sheep was under general anesthesia. Figure from Wang et al. [98].

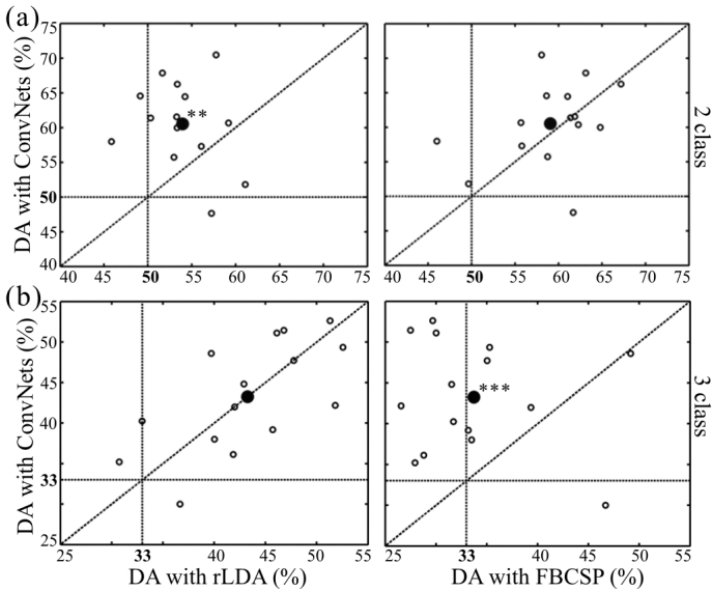



**Figure 9.7: Results of decoding auditory evoked responses from sheep.** rlDA, FBSCP and the deep ConvNet were the decoding models. Open circles represent accuracies for individual experiment days and closed circles represent the average over these accuracies. Figure from Wang et al. [98].

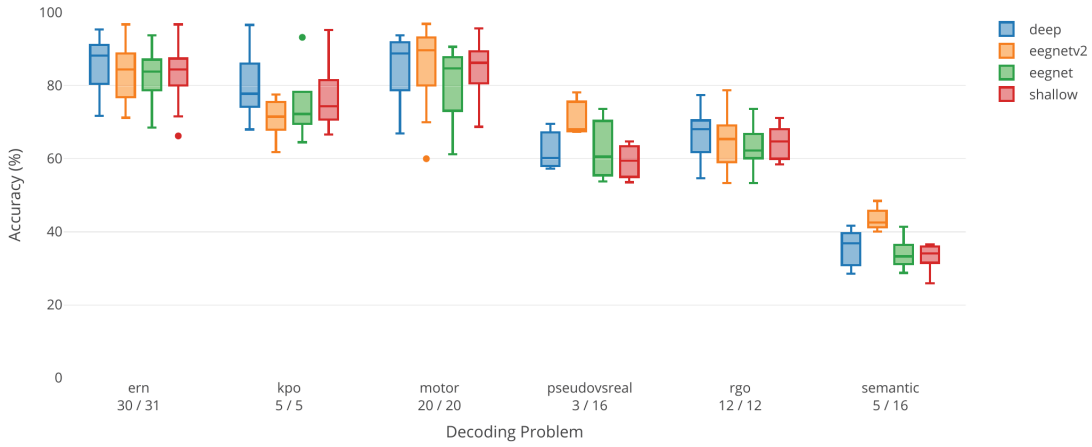



**Figure 9.8: Per-dataset results for the large-scale evaluation of deep ConvNet, shallow ConvNet and two versions of EEGNet.** Boxplots show the distribution over per-subject accuracies for the individual decoding tasks. ern, kpo and rgo are the error-related datasets, ern: Error-related negativity Eriksen flanker task, KPO: KUKA Pouring Observation paradigm, rgo: robot-grasping observation paradigm. motor is the high-gamma dataset with 6 additional subjects that were excluded for data quality reasons from [77]. pseudovsreal and semantic are two semantic processing datasets to classify silent repetitions of pseudowords vs. realwords (pseudovsreal) or different semantic categories (semantic) . Figure from Heilmeyer et al. [33].

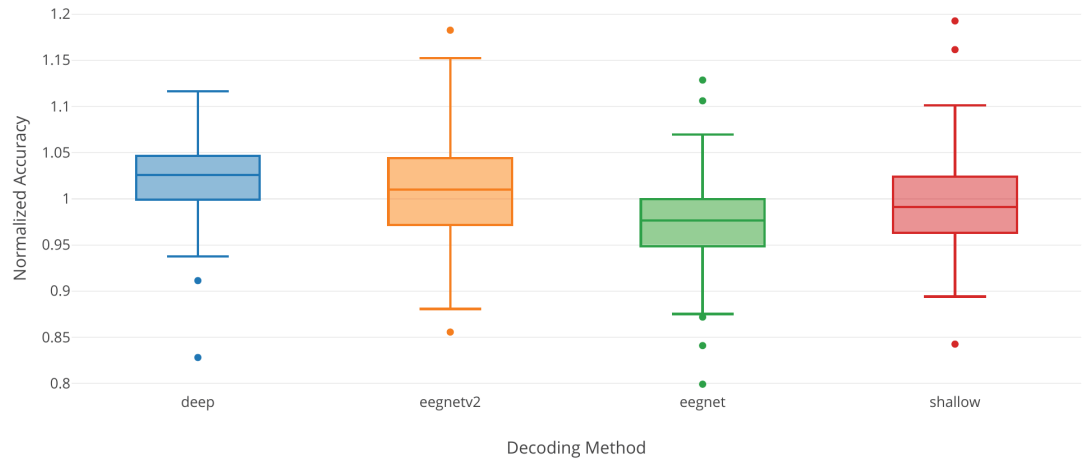



**Figure 9.9: Dataset-averaged results for the large-scale evaluation of deep ConvNet, shallow ConvNet and two versions of EEGNet.** Accuracies are normalized to the average of the accuracies of all models. Figure from Heilmeyer et al. [33].

# 10 DECODING PATHOLOGY

ConvNets diagnose pathology with good accuracy even from very short amounts of time

- ConvNets reach around 85
- Can reach high accuracies using a single minute per recording during inference
- Struggle with recordings where contextual features like age and sleep affect the doctors diagnosis
- Seem to learn temporal slowing indicates pathology, strong occipital alpha indicates healthy

We also evaluated our deep ConvNets for automatic medical diagnosis from EEG. EEG is important in clinical practice both as a screening method as well as for hypothesis-based diagnostics, e.g., in epilepsy or stroke. One of the main limitations of using EEG for diagnostics is the required time and specialized knowledge of experts that need to be well-trained on EEG diagnostics to reach reliable results. Therefore, a deep-learning approach that aids in the diagnostic process could make EEG diagnosis more widely accessible, reduce time and effort for clinicians and potentially make diagnoses more accurate.

Text and figures in this section are adapted from Schirrmeister et al. [75]. I was the main contributor to Schirrmeister et al. [75].

## 10.1 DATASET AND PREPROCESSING

### 10.1.1 Temple University Hospital EEG Abnormal Corpus

We used the Temple University Hospital (TUH) EEG Abnormal Corpus for evaluating our deep ConvNets on pathology detection from EEG. The Temple University Hospital (TUH) EEG Abnormal Corpus 1.1.2 is a dataset of manually labeled normal and pathological clinical EEG recordings. It is taken from the TUH EEG Data Corpus which contains over 16000 clinical recordings of more than 10000 subjects from over 12 years [**obeid_temple_2016**]. The Abnormal Corpus contains 3017 recordings, 1529 of which were labeled normal and 1488 of which were labeled pathological. The

| | | FILES | PATIENTS |
|---|---|---|---|
| Train | Normal | 1379 (50%) | 1238 (58%) |
| | Pathological | 1361(50%) | 894 (42%) |
| | Rater Agreement | 2704 (99%) | 2107 (97%) |
| | Rater Disagreement | 36 (1%) | 25 (0%) |
| Evaluation | Normal | 150 (54%) | 148 (58%) |
| | Pathological | 127 (46%) | 105 (42%) |
| | Rater Agreement | 277 (100%) | 253 (100%) |
| | Rater Disagreement | 0 (0%) | 0 (0%) |

**Table 10.1: TUH EEG Abnormal Corpus 1.1.2 Statistics.** Obtained from https://www.isip.piconepress.com/projects/tuh_eeg/. Rater agreements refer to the agreement between the student annotator of the file and the medical report written by a certified neurologist.

Corpus was split into a training and evaluation set, see Table 10.1. Recordings were acquired from at least 21 standard electrode positions and with a sampling rate of in most cases 250 Hz. Per recording, there are around 20 minutes of EEG data. The inter-rater agreement on between the medical report of a certified neurologist and a medical student annotator was 99% for the training recordings and 100% for the evaluation recordings, also see Table 10.1.

### 10.1.2 Preprocessing

We minimally preprocessed the data with these steps: 1. Select a subset of 21 electrodes present in all recordings. 2. Remove the first minute of each recording as it contained stronger artifacts. 3. Use only up to 20 minutes of the remaining recording to speed up the computations. 4. Clip the amplitude values to the range of $\pm 800\ \mu V$ to reduce the effects of strong artifacts. 5. Resample the data to 100 Hz to further speed up the computation.

### 10.1.3 Decoding from Reduced EEG Time Segments

We also evaluated the ConvNets on reduced versions of the datasets, using only the first 1, 2, 4, 8, or 16 minutes after the first minute of the recording (the first minute of the recordings was always excluded because it appeared to be more prone to artifact contamination than the later time windows). We reduced either only the training data, only the test data, or both. These analyses were carried out to study how long

EEG recordings need to be for training and for predicting EEG pathologies with good accuracies.

## 10.2 NETWORK ARCHITECTURES

We used our deep and shallow ConvNets with only minor modifications to the architecture. To use larger time windows to make a single prediction, we adapted the architectures by changing the final layer kernel length so the ConvNets have an input length of about 600 input samples, which correspond to 6 seconds for the 100 Hz EEG input. Additionally, we moved the pooling strides of the deep ConvNet to the convolutional layers directly before each pooling. This modification, which we initially considered a mistake, allowed us to grow the ConvNet input length without strongly increased computation times and provided good accuracies in preliminary experiments on the training data; therefore we decided to keep it.

## 10.3 NETWORK TRAINING

As in other studies, we optimized the ConvNet parameters using stochastic gradient descent with the optimizer Adam [**kingma_adam:_2014**]. To make best use of the available data, we trained the ConvNets on maximally overlapping time crops using cropped training as described in Chapter 5. Code to reproduce the results of this study is available under https://github.com/robintibor/auto-eeg-diagnosis-example.

## 10.4 AUTOMATIC ARCHITECTURE OPTIMIZATION

We also carried out a preliminary study of automatic architecture optimization to further improve our ConvNet architectures. To that end, we used the automatic hyperparameter optimization algorithm SMAC [**hutter_sequential_2011**] to optimize architecture hyperparameters of the deep and shallow ConvNets, such as filter lengths, strides and types of nonlinearities. As the objective function to optimize via SMAC, we used 10-fold cross-validation performance obtained on the first 1500 recordings of the training data (using each fold as an instance for SMAC to speed up the optimization). We set a time limit of 3.5 hours for each configuration run on a single fold. Runs that timed out or crashed (e.g., networks configurations that did not fit in GPU memory) were scored with an accuracy of 0%.

## 10.5 DEEP AND SHALLOW CONVNETS REACHED STATE-OF-THE-ART RESULTS

| | ACCURACY | SENSITIVITY | SPECIFICITY | CROP-ACCURACY |
|---|---|---|---|---|
| Baseline [21] | 78.8 | 75.4 | 81.9 | n.a. |
| Deep | 85.4 | 75.1 | 94.1 | 82.5 |
| Shallow | 84.5 | 77.3 | 90.5 | 81.7 |
| Linear | 51.4 | 20.9 | 77.3 | 50.2 |

**Table 10.2: Decoding accuracies for discriminating normal and pathological EEG with deep and shallow ConvNets.** For deep and shallow ConvNets, mean over five independent runs with different random seeds. Deep and shallow ConvNet outperformed the feature-based deep learning baseline. Note that the baseline was evaluated on an older version of the corpus that has since been corrected to not contain the same patient in training and test recordings among other things. Results from Schirrmeister et al. [75].

Both the deep and the shallow ConvNet outperformed the only results that had been published on the TUH Abnormal EEG Corpus at the time (see Table 10.2). Both ConvNets were more than 5% better than the baseline method of a convolutional network that included multiple fully connected layers at the end and took precomputed EEG features of an entire recording as one input [21].The ConvNets as applied here reduced the error rate from about 21% to about 15%. We also tested a linear classifier on the same 6-second inputs as our ConvNets. The linear classifier did not reach accuracies substantially different from chance (51.4%).

Interestingly, both of our ConvNet architectures already reached higher accuracies than the baseline when evaluating single predictions from 6-second crops. The average per-crop accuracy of individual predictions was only about 3% lower than average per-recording accuracy (averaged predictions of all crops in a recording). Furthermore, the individual prediction accuracies were already about 3% higher than the per-recording accuracies of the baseline. This implies that predictions with high accuracies can be made from just 6 seconds of EEG data.

Both of our ConvNets made more errors on the pathological recordings, as can be seen from Figure 10.1. Both ConvNets reached a specificity of above 90% and a sensitivity of about 75-78%. Confusion matrices between both approaches were very similar. Relative to the baseline, they reached a similar sensitivity (0.3% smaller for the deep ConvNet, 1.9% higher for the shallow ConvNet), and a higher specificity (12.2% higher for the deep ConvNet and 8.6% higher for the shallow ConvNet).

Deep ConvNets already reached their best trialwise accuracies with only one minute of data used for the prediction. While the reduction of the amount of length of the

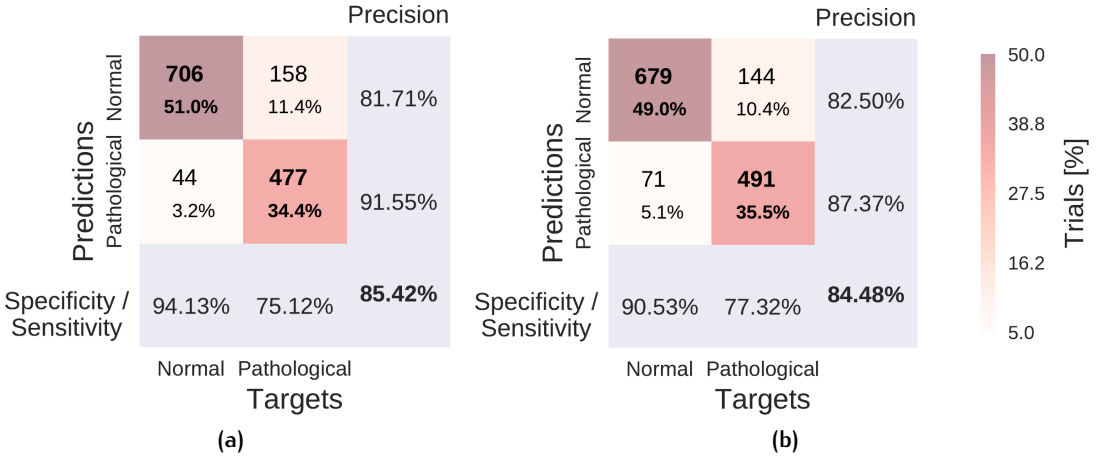



**Figure 10.1: Confusion Matrices for deep and shallow ConvNets**, summed over five independent runs. Each entry of row r and column c for upper-left 2x2-square: Number of trials of target r predicted as class c (also written in percent of all trials). Bold diagonal corresponds to correctly predicted trials for both classes. Percentages and colors indicate fraction of trials in each cell relative to all trials. The lower-right value: overall accuracy. The first two values in the bottom row correspond to sensitivity and specificity. Rightmost column corresponds to precision defined as the number of trials correctly predicted for class r/number of trials predicted as class r. Figure from Schirrmeister et al. [75].

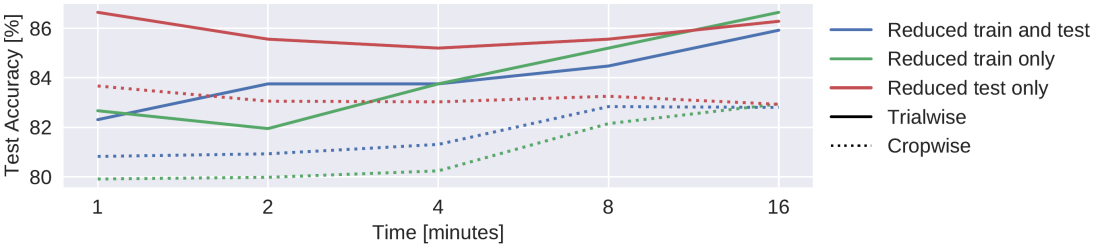



**Figure 10.2: Results on reduced datasets for deep ConvNet.** Train and/or test (evaluation) dataset was reduced from 20 minutes per recording to 1,2,4,8, or 16 minutes per recording, results are shown on the test set. Notably, when only reducing the duration of the test set recordings, maximal accuracies were observed when using just 1 minute. We note that these results are each based on one run only; the slightly better performance than in Table 10.2 may thus be due to noise. Figure from Schirrmeister et al. [75].

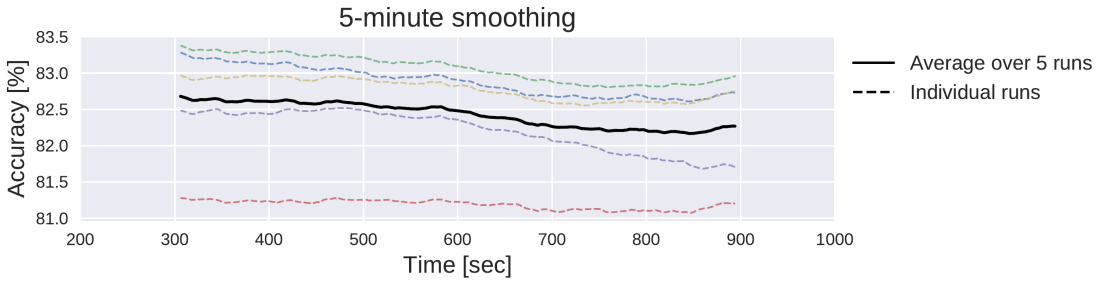



**Figure 10.3: Moving average of cropwise accuracies for the deep ConvNet.** 5-minute moving averages of the cropwise accuracies of the deep ConvNet, averaged over all test set recordings. Dashed lines represent 5 individual training runs with different random seeds, solid black line represents mean over results for these runs. x-axis shows center of 5-minute averaging window. Figure from Schirrmeister et al. [75].

training data led to crop- and trialwise accuracy decreases on the test data, reductions in the test data did not have such an effect (see Figure 10.2). Remarkably, both crop- and trialwise accuracies slightly decreased when going from 1 minute to 2 or 4 minutes of test data. To investigate whether earlier parts of the recordings might be more informative, we also computed a 5-minute moving average of the cropwise accuracies on the test data for the Deep ConvNet trained on the full data. We show the average over all recordings for these moving averages in (see Figure 10.3). Noticeably, as expected, accuracies slightly decreased with increasing recording time. However, the decrease is below 0.5% and thus should be interpreted cautiously.

## 10.6 ARCHITECTURE OPTIMIZATION YIELDED UNEXPECTED NEW MODELS

The models discovered by automated architecture optimization were markedly different from our original deep and shallow ConvNets. For example, the optimized architectures used only 1.8 and 3.7 seconds of EEG data for the optimized deep and shallow ConvNet, respectively, in contrast to about 6 seconds in the original versions. While the improved performance of these modified architectures for the 10-fold cross-validation on the training dataset (2.1% and 1.4% improvement for deep and shallow ConvNets, respectively) did not generalize to the evaluation set (0.9% and 1.5% deterioration for deep and shallow ConvNets, respectively, see Table 10.3), the modifications to the original network architectures already provided interesting insights for further exploration: For example, in the case of the shallow ConvNet, the modified architecture did not use any of the original nonlinearities, but used max

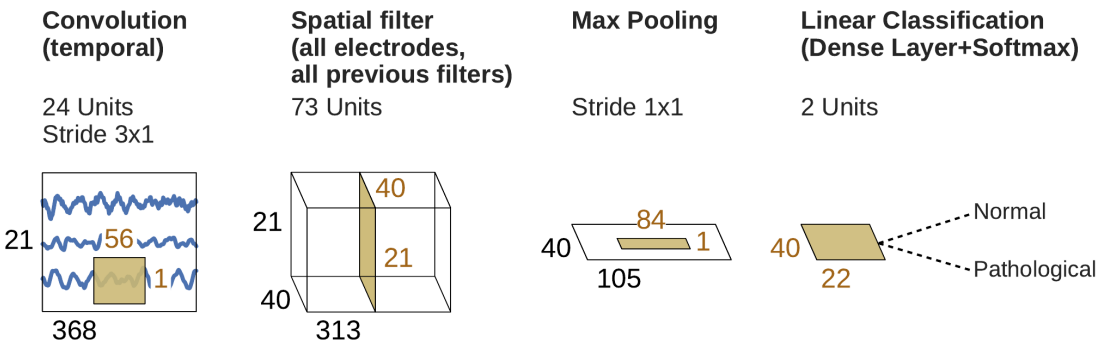



**Figure 10.4: Final shallow ConvNet architecture selected by SMAC.** Conventions as in Fig. **??**. Note that max pooling is the only nonlinearity SMAC decided to use. Figure from Schirrmeister et al. [75].

| | ARCHITECTURE CONFIGURATION | TRIAL | CROP | TRIAL | CROP |
|---|---|---|---|---|---|
| **Deep** | Default | 84.2 | 81.6 | 85.4 | 82.5 |
| | Optimized | 86.3 | 80.9 | 84.5 | 81.3 |
| **Shallow** | Default | 84.5 | 82.1 | 84.5 | 81.7 |
| | Optimized | 85.9 | 80.3 | 83.0 | 79.8 |

**Table 10.3: Decoding accuracies with the default of deep and shallow ConvNets as well as versions optimized by automatic architecture optimization.** Train here refers to 10-fold cross-validation on the 1500 chronologically earliest recordings of the training data. . Results from Schirrmeister et al. [75].

pooling as the only nonlinearity (see Fig. **??**), a configuration we had not considered in our manual search so far.

## 10.7 VISUALIZATION

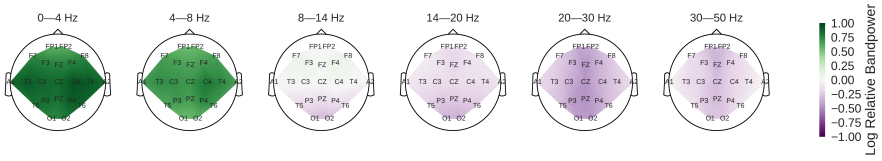
We analyzed the spectral power changes in the data itself and the spectral characteristics of the function the deep networks learned on the data.

To understand class-specific spectral characteristics in the EEG recordings, we analyzed band powers in five frequency ranges: delta (0–4 Hz), theta (4–8 Hz), alpha (8–14 Hz), low beta (14–20 Hz), high beta (20–30 Hz) and low gamma (30–50 Hz).

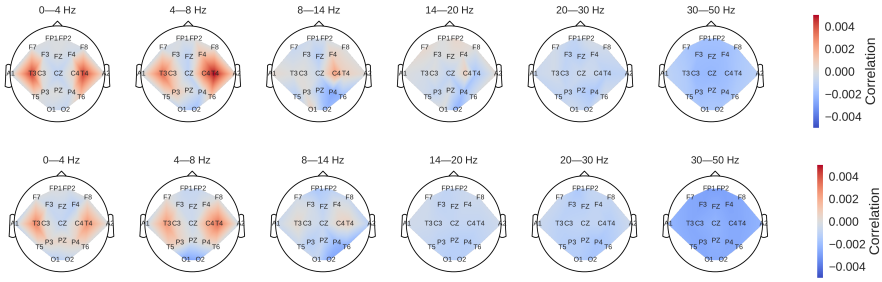
For this, we performed the following steps: 1. Compute a short-term Fourier transformation with window size 12 seconds and overlap 6 seconds using a Blackman-Harris window. 2. Compute the median over all band powers of all windows and recordings in each frequency bin; independently for pathological and normal recordings. 3.

Compute the log ratio of these median band powers of the pathological and normal recordings. 4. Compute the mean log ratio over all frequency bins in each desired frequency range for each electrode. 5. Visualize the resulting log ratios as a topographical map.

To better understand the spectral characteristics of the function the ConvNets learned used in this study, we also used the perturbation-based visualization method described in Schirrmeister et al. [77].




**(a)** Pathological vs. normal relative spectral bandpower differences for the training set. Shown is the logarithm of the ratio of the median bandpower of the pathological vs. normal (according to the experts' ratings) EEG recordings.




**(b)** Input-perturbation network-prediction correlation maps for the deep (top) and shallow (bottom) ConvNet. Correlation of predictions for the pathological class with amplitude perturbations. Scalp maps revealed for example a bilateral positive correlation for the delta and theta frequency ranges and a spatially more broadly distributed negative correlation for the beta and low gamma frequency ranges, indicating that the ConvNets used these frequency components in their decisions

**Figure 10.5: Spectral power differences and input-perturbation network-prediction correlation maps.** Figure from Schirrmeister et al. [75].

Power was broadly increased for the the pathological class in the low frequency bands (delta and theta range) and decreased in the beta and low gamma ranges (Figure 10.5a). Alpha power was decreased for the occipital electrodes and increased for more frontal electrodes.

Scalp maps of the input-perturbation effects on predictions for the pathological class for the different frequency bands showed effects consistent with the power spectra in (Figure 10.5b). Both networks strongly relied on the lower frequencies in the delta and theta frequency range for their decoding decisions.

## 10.8 ANALYSIS OF WORD FREQUENCIES IN THE MEDICAL REPORTS

Furthermore, to better understand what kind of recordings are easier or harder for the ConvNets to correctly decode, we analyzed the textual clinical reports of each recording as included in the TUH Abnormal EEG Corpus. Specifically, we investigated which words were relatively more or less frequent in the incorrectly predicted recordings compared with the correctly predicted recordings. We performed this analysis independently for both the normal and the pathological class of recordings. Concretely, for each class, we first computed the relative frequencies $f_{i-}$ for each word $w_{i-}$ in the incorrectly predicted recordings, i.e.: $f_{i-} = \frac{|w_{i-}|}{\sum_i |w_{i-}|}$, where $|w_{i-}|$ denotes the number of occurrences for word $w_i$ in the incorrectly predicted recordings. We then computed the frequencies $f_{i+}$ in the same way and computed the ratios $r_i = f_{i-}/f_{i+}$. Finally, we analyzed words with very large ratios ($\gg 1$) and very small ratios ($\ll 1$) by inspecting the contexts of their occurrences in the clinical reports. This allowed us to gain insights into which clinical/contextual aspects of the recordings correlated with ConvNets failures.

Most notably, `small` and `amount` had a much larger word frequency (15.5 times larger) in the incorrectly predicted pathological recordings compared with the correctly predicted pathological recordings. Closer inspection showed this is very sensible, as `small amount` was often used to describe more subtle EEG abnormalities (`small amount of temporal slowing`, `Small amount of excess theta`, `Small amount of background disorganization`, `A small amount of rhythmic, frontal slowing`), as this subtlety of changes was likely the cause of the classification errors.

Secondly, other words with a notably different frequency were `age` (9.7 times larger) and `sleep` (3 occurrences in 630 words of texts of incorrectly predicted recordings, not present in texts of correctly predicted recordings). Both typically indicate the clinician used the age of the subject or the fact that they were (partially) asleep during the recording to interpret the EEG (`Somewhat disorganized pattern for age`, `Greater than anticipated disorganization for age.`, `A single generalized discharge noted in stage II sleep.`). Obviously, our ConvNets trained only on EEG do not have access to this context information, leaving them at a disadvantage compared to the clinicians and highlighting the potential of including contextual cues such as age or vigilance in the training/decoding approach.

Inspection of the textual records of misclassified normal recordings did not provide much insight, as they are typically very short (e.g., `Normal EEG.`, `Normal EEG in wakefulness.`).

Finally, consistent with the strong usage of the delta and theta frequency range by the ConvNets as seen in the input-perturbation network-prediction correlation maps (Figure 10.5b), `slowing` and `temporal` are the 6th and 10th most frequently occurring words in the textual reports of the pathological recordings, while never occurring

in the textual reports of the normal recordings (irrespective of correct or incorrect predictions).

## 10.9 DISCUSSION

To the best of our knowledge, the ConvNet architectures used in this study achieved the best accuracies published so far on the TUH EEG Abnormal Corpus. The architectures used were only very slightly modified versions of ConvNet architectures that we previously introduced to decode task-related information. This suggests that these architectures might be broadly applicable both for physiological and clinical EEG. The identification of all-round architectures would greatly simplify the application of deep learning to EEG decoding problems and expand their potential use cases.

Remarkably, the ConvNets already reached good accuracies based on very limited time segments of the EEG recordings. Further accuracy improvements could thus be possible with improved decoding models that can extract and integrate additional information from longer timescales. The exact nature of such models, as well as the amount of EEG they would require, remains to be determined. More accurate decoding models could either be ConvNets that are designed to intelligently use a larger input length or recurrent neural networks, since these are known to inherently work well for data with information both on shorter and longer term scales. Furthermore, combinations between both approaches, for example using a recurrent neural network on top of a ConvNet, as they have been used in other domains like speech recognition [45, 71, 72], are promising.

Our automated architecture optimization provided interesting insights by yielding configurations that were markedly different from our hand-engineered architectures, yet reached similar accuracies. Since the marked improvements in training performance did not improve the evaluation accuracies in this study, in future work, we plan to use more training recordings in the optimization and study different cross-validation methods to also improve evaluation accuracies. A full-blown architecture search [52, 53, 68, 104, 105] could also further improve accuracy. With such improved methods it would also be important not only to decode pathological vs. normal EEG in a binary fashion, but to also evaluate the possibility to derive more fine-grained clinical information, such as the type of pathological change (slowing, asymmetry, etc) or the likely underlying disorder (such as epilepsy).

Any of these or other improvements might eventually bring the machine-learning decoding performance of pathological EEG closer to human-level performance. Since clinicians make their judgments from patterns they see in the EEG and other available context information, there is no clear reason why machine learning models with access to the same information could not reach human-level accuracy. This human-level performance is a benchmark for decoding accuracies that does not exist for other brain-

signal decoding tasks, e.g. in decoding task-related information for brain-computer interfaces, where there is inherent uncertainty what information is even present in the EEG and no human-level benchmark exists.

Our perturbation visualizations of the ConvNets' decoding behavior showed that they used spectral power changes in the delta (0-4 Hz) and theta (4-8 Hz) frequency range, particularly from temporal EEG channels, possibly alongside other features (**??**). This observation is consistent both with the expectations implied by the spectral analysis of the EEG data (Figure 10.5a) and by the textual reports that frequently mentioned `temporal` and `slowing` with respect to the pathological samples, but never in the normal ones. Our perturbation visualization showed results that were consistent with expectations that the ConvNets would use the bandpower differences between the classes that were already visible in the spectra to perform their decoding. Similarly, the textual reports also yielded plausible insights, e.g., that `small amounts` of abnormalities as indicated in the written clinical reports were more difficult for the networks to decode correctly. Additionally, inspection of the textual reports also emphasized the importance of integrating contextual information such as the age of the subject.

Still, to yield more clinically useful insights and diagnosis explanations, further improvements in ConvNet visualizations are needed. Deep learning models that use an attention mechanism might be more interpretable, since these models can highlight which parts of the recording were most important for the decoding decision. Other deep learning visualization methods like recent saliency map methods [38, 56] to explain individual decisions or conditional generative adversarial networks [55, 83] to understand what makes a recording pathological or normal might further improve the clinical benefit of deep learning methods that decode pathological EEG.

Another option for more interpretable networks we explore in the next chapter of this thesis are invertible networks, neural networks that are designed to be invertible, see Chapter 7 for methods and Chapter 11 for results.

Open Questions

- How to further visualize the features networks learn to diagnose pathology?

# 11 UNDERSTANDING PATHOLOGY DECODING WITH INVERTIBLE NETWORKS

EEG-InvNet and EEG-CosNet can reveal learned features for EEG pathology decoding

- EEG-InvNet can reach 85.5
- Visualizations show networks learn well-known features like temporal slowing or occipital alpha
- Visualizations also reveal surprising learned features in the very low frequencies up to 0.5 Hz

After our initial work on pathology decoding, we wanted to gain a deeper understanding of the features deep networks learn to distinguish healthy from pathological recordings. For that, we used invertible networks as generative classifiers since they offer more ways to visualize their learned prediction function in input space. Our EEG-InvNet reached competitive accuracies on the pathology decoding task. We visualize prototypes of the two classes as well as individual electrode signals predictive of a certain class independent of the signals at other electrodes. These visualizations revealed both well-known features like temporal slowing or occipital alpha as well as surprising patterns in the very low frequencies (<= 0.5 Hz). To gain an even better understanding, we distilled the invertible network's knowledge into a very small network called EEG-CosNet that is interpretable by design. These visualizations showed regular patterns in the alpha and beta range associated with healthy recordings and a diverse set of more irregular waveforms associated with pathology. For the very low frequencies, visualizations revealed a frontal component predicting the healthy class and other components with spatial topographies including the temporal areas predicting the pathological class.

All work presented in this chapter is novel unpublished work performed by me in the context of this thesis.

## 11.1 DATASET, TRAINING DETAILS AND DECODING PERFORMANCE

We apply our EEG-InvNet to pathology decoding on the same TUH dataset as in Chapter 10. We use only 2 minutes of each recording at 64 Hz, and input 2

| DEEP | SHALLOW | TCN | EEGNET | EEG-INVNET |
|---|---|---|---|---|
| 84.6 | 84.1 | 86.2 | 83.4 | 85.5 |

**Table 11.1: Accuracy of EEG-InvNet on pathology decoding.** Accuracies of regular ConvNets taken from Gemein et al. [25].

seconds as one example to the invertible network. This reduced dataset allows fast experimentation while still yielding good decoding performance. We used AdamW [48] as our optimizer and cosine annealing with restarts [47] every 25 epochs as our learning rate schedule. We emphasize these details were not heavily optimized for maximum decoding performance, but rather chosen to obtain a robustly performing model worth investigating more deeply. Results in Table 11.1 show that our EEG-InvNet compares similar than regular ConvNets, even better than some ConvNets, therefore motivating a deeper investigation into its learned features.

## 11.2 CLASS PROTOTYPES

Class prototypes reveal known oscillatory features and surprisingly hint at the use of very-low-frequency information by the invertible network. We inverted the learned latent means of the healthy and the pathological class distributions back to the input space to visualize the most likely healthy and most likely pathological examples under the learned distribution, see also Section 7.5. Visualizations in Figure 11.1 show differences in the alpha rhythm like a stronger alpha rhythm at O1 in the healthy example. We also see further differences with a variety of different oscillatory patterns present for both classes. Surprisingly, there are also differences in the very low frequencies like substantially different mean values for FP1 and FP2 for the two class prototypes, which we will further investigate later. One challenge of this visualization is that one has to look at each prototype as one complete example and cannot interpret signals at individiual electrodes independently. This is what we tackle in our next visualization.

## 11.3 PER-ELECTRODE PROTOTYPES

The per-channel prototypes reveal interesting learned features for the two classes (see Figure 11.2). The pathological prototypes show strong low-frequency activity, for example at T3 and T4, consistent with slowing as a biomarker for pathology. The healthy signal shows alpha activity, for example at C4 and T6. Besides these patterns, a lot of other interesting patterns may be interesting to further investigate. One of

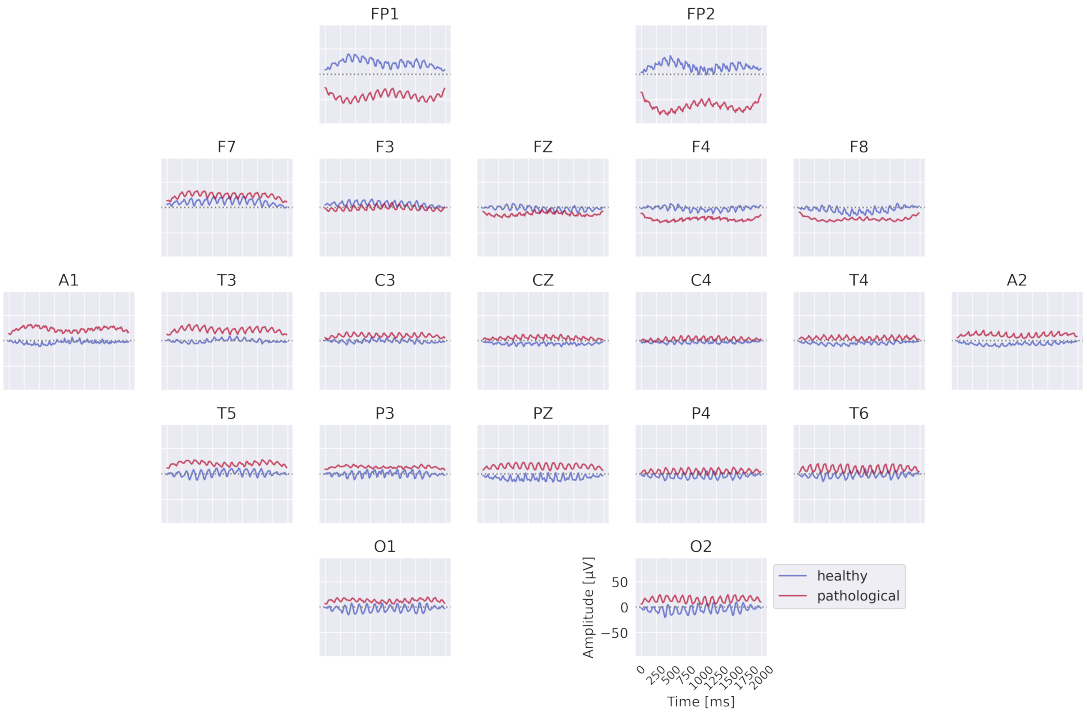



**Figure 11.1: Learned class prototypes from EEG-InvNet.** Obtained by inverting learned means of class-conditional gaussian distributions from latent space to input space through the invertible network trained for pathology decoding.

them, the differences in the very low frequencies will be further explored below. Note that it was not possible to synthesize a signal that is clearly indicative of one class independent of the other electrodes for all electrodes. This is to be expected if the EEG-InvNet uses a feature inherently impossible to recreate within a single electrode like the degree of synchrony between signals at different electrodes.

## 11.4 EEG-COSNET VISUALIZATIONS

| | EEG-INVNET PREDICTIONS | ORIGINAL LABELS |
|---|---|---|
| Train | 92.5 | 89.1 |
| Test | 88.8 | 82.6 |

**Table 11.2: Accuracy of EEG-CosNet on invertible network predictions and original labels.**

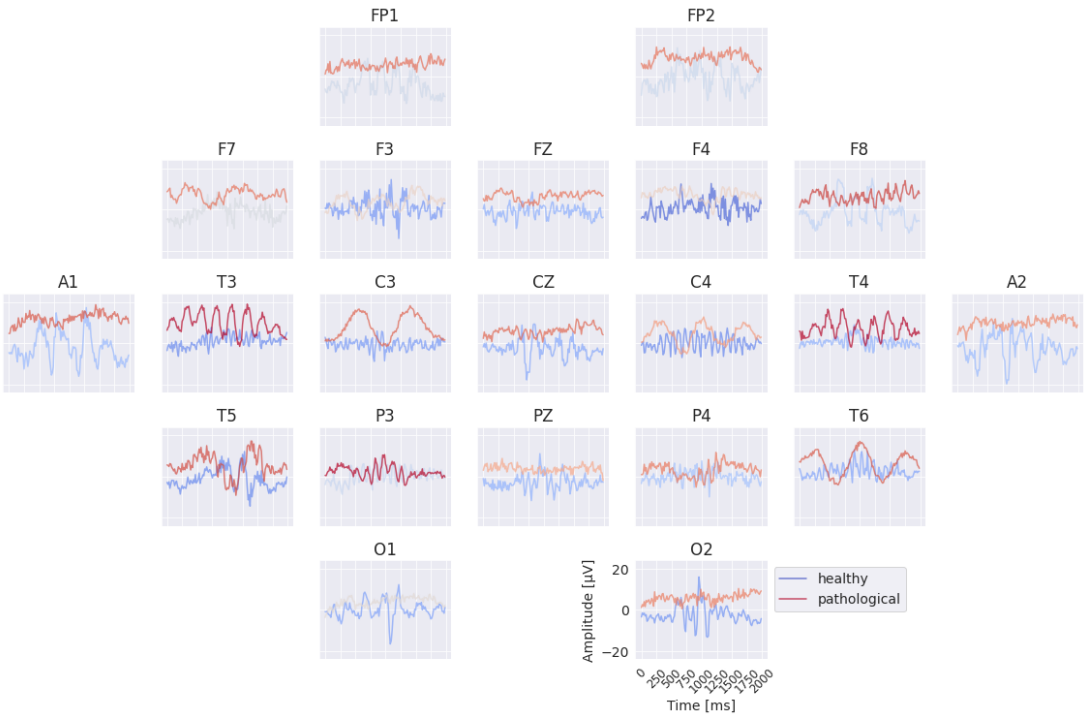



**Figure 11.2: Learned per-electrode prototypes from EEG-InvNet.** Each electrodes' input is optimized independently to increase the invertible networks prediction for the respective class. During that optimization, signals for the other non-optimized electrodes are sampled from the training data. Color indicates average softmax prediction over 10000 samples for the other electrodes. Very prominent slowing patterns appear for the pathological class at multiple electrodes.

Results for the EEG-CosNet show that a large fraction of the predictions of the invertible network can be predicted from a relatively small number of mostly neurophysiologically plausible spatio-temporal patterns. EEG-CosNet predicts 88.8% of the recordings in the same way as the EEG-InvNet and retains a test set label accuracy of 82.6% (see Table 11.2. This shows that from just 64 spatiotemporal features, the EEG-CosNet is able to predict the vast majority of the EEG-InvNet predictions. Still, the remaining gap indicates that the EEG-InvNet has learned some features that the EEG-CosNet cannot represent.

Visualizations in **??** show more regular waveforms in the alpha and beta-frequency ranges with higher association for the healthy class and more waveforms in other frequency ranges as well as less regular waveforms with higher association for the pathological class. As examples for the healthy class, plots 1 and 3 show oscillations with a strong alpha component and plots 15-17 show oscillations with strong beta components. For the pathological class, we see slower oscillations, e.g., in plots 53 and 60, and also more irregular waveforms in, e.g., plots 49 and 52.

## 11.5 INVESTIGATION OF VERY LOW FREQUENCIES

One surprising observation from the visualizations are differences in the very low frequencies (<=0.5 Hz) between the two class prototypes. For example, the very different mean values in the class prototypes for FP1 and FP2 suggest very low frequency information differs between the two classes on those electrodes. These kinds of differences motivated us to more deeply investigate in how far very low frequency information is predictive of pathology.

| EEG-INVNET | EEG-COSNET | FOURIER-GMM |
|---|---|---|
| 75.4 | 75.0 | 75.4 |

**Table 11.3: Test accuracy on data lowpassed below 0.5 Hz.**

For this, we trained an EEG-InvNet on data lowpassed to be below 0.5 Hz. For the lowpass, we first removed all Fourier components above 0.5 Hz for each recording and also for each 2-second input window for the network. This retained 75.4% accuracy with the EEG-InvNet, indicating even these very low frequencies remain fairly informative about the pathologicality of the recording. We additionally trained the EEG-CosNet with a temporal filter spanning the entire input window length of 2 seconds and found it to retain 75% test accuracy. Finally, we also directly trained a 8-component gaussian mixture model Fourier-GMM in Fourier space. Only 3 dimensions per electrode remain: real value of the 0-Hz component ( summed values of the input window) and real and imaginary value of the 0.5-Hz Fourier component). Each of the 8 mixture components had learnable class weights, how much each mixture component contributed to that classes learned distribution. The Fourier-GMM also retains 75.4% test accuracy. All results are shown in Table 11.3.

### 11.5.1 EEG-InvNet Visualizations

The visualizations of the EEG-InvNet show several interesting features. The class prototypes in Figure 11.4 show differences at most electrodes, especially pronounced for A1 and A2. The per-electrode prototypes in Figure 11.5 show predictive information in the T3,T4 and T6 electrodes.

### 11.5.2 EEG-CosNet Visualizations

Visualization of the EEG-CosNet in Figure 11.6 contain strong frontally components associated with the healthy class and components in temporal areas associated with the pathological class. The temporal components are in line with the per-electrode

visualization, and the frontal components were already visible as differences in mean signal values in the class prototypes on the original data.

### 11.5.3 Fourier-GMM Visualizations

Open Questions

- What other methods may in the future help understand discriminative features in the EEG signal?

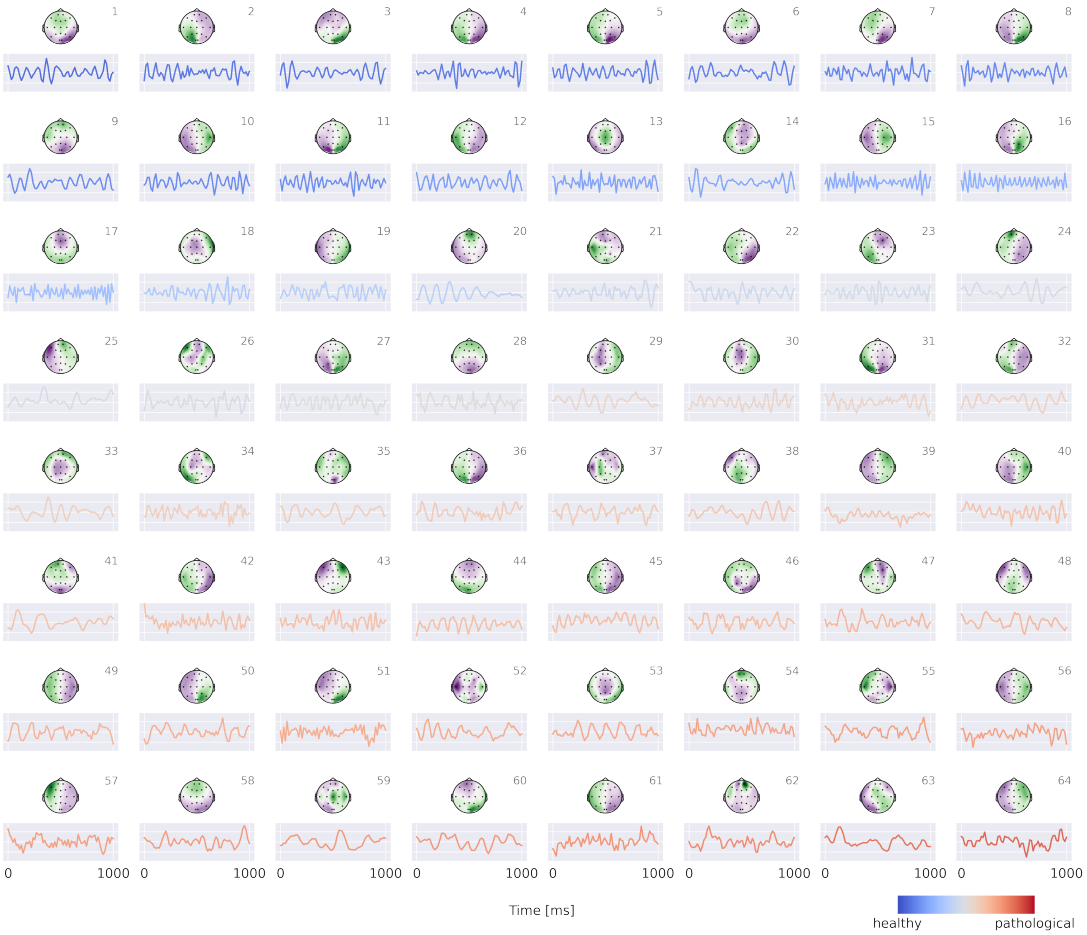



Figure 11.3: **Visualization of small interpretable EEG-CosNet trained to mimic the EEG-InvNet.** Scalp Plots are spatial filter weights transformed to patterns, signals below each scalp plot show corresponding convolutional filter. Signal colors represent the weights of the linear classification layer, transformed to patterns (see Section 7.7 for an explanation). Plots are sorted by these colors. Note that polarities of the scalp plots and temporal waveforms are arbitrary as absolute cosine similarities are computed on the spatially filtered and temporally convolved signals.

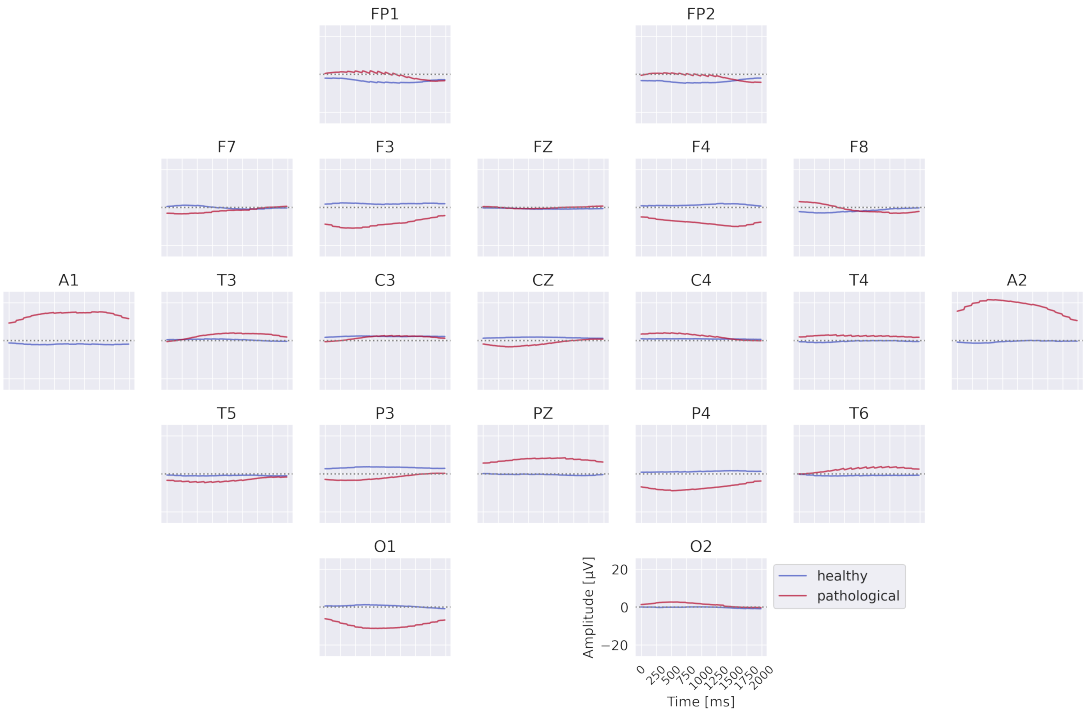



**Figure 11.4: Class prototypes for the EEG-InvNet trained on data lowpassed to be below 0.5 Hz.** Note large differences at A1 and A2.

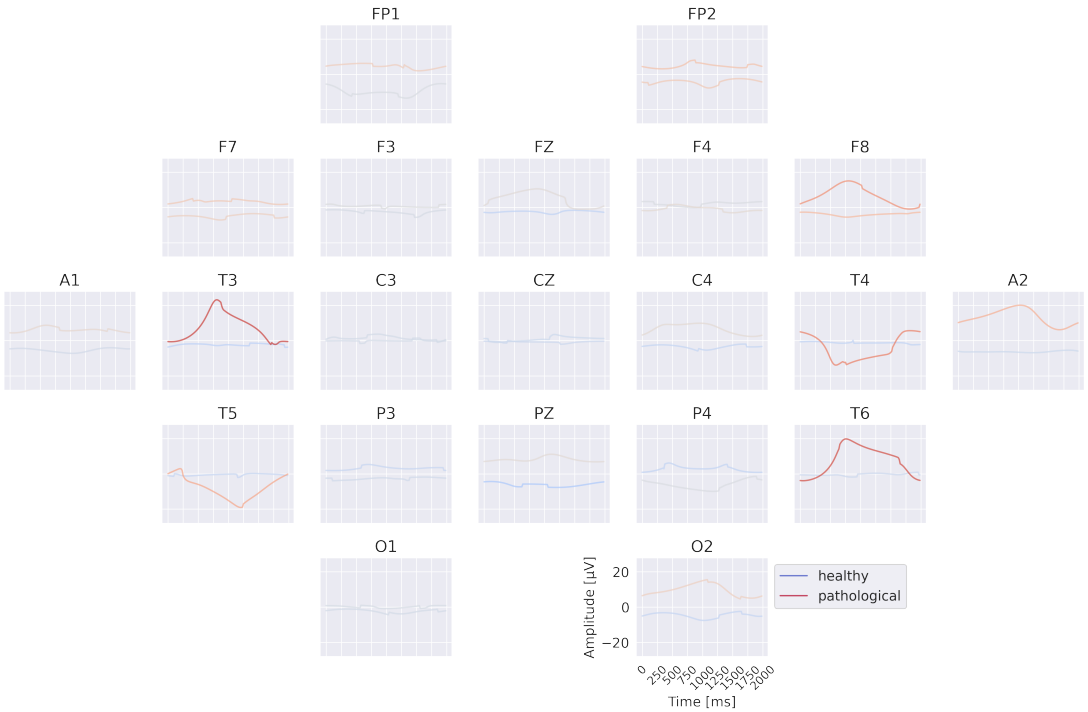



**Figure 11.5: Per-electrode prototypes for EEG-InvNet trained on data lowpassed below 0.5 Hz.** Note strongly predictive signals at T3,T4,T6.

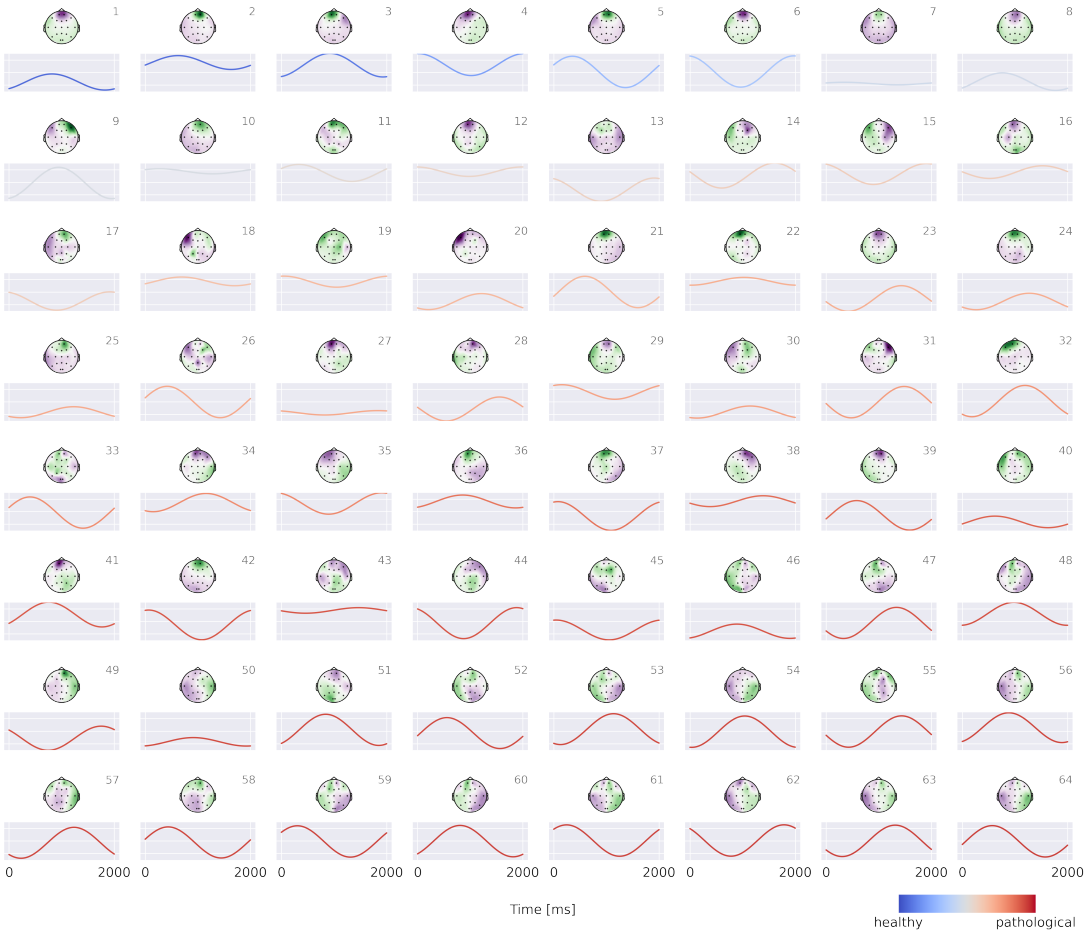



**Figure 11.6: Spatiotemporal patterns for EEG-CosNet trained on lowpassed data below 0.5 Hz.** Note large frontal components associated with healthy class.

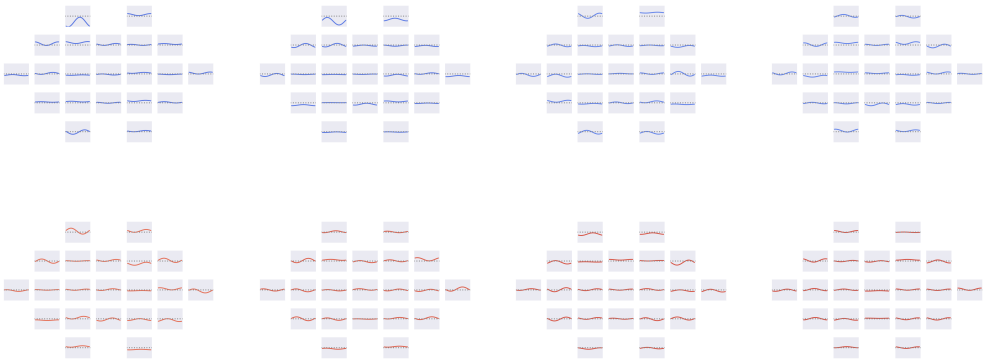


Figure 11.7: **Means of the Fourier-GMM mixture components shown after inversion into input space.** Note clearly visible frontal signals in the components for the healthy class.

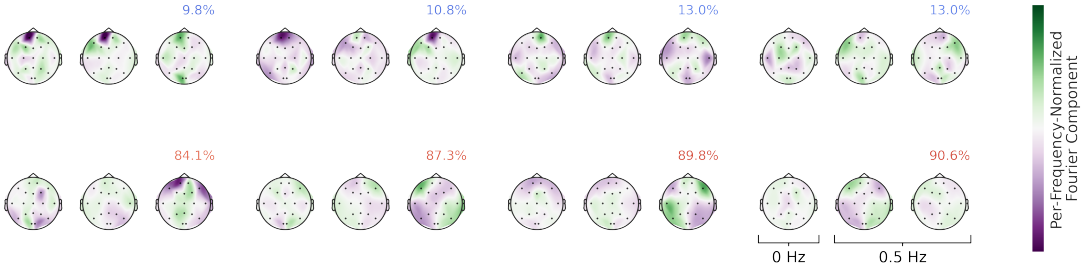



Figure 11.8: **Means of the Fourier-GMM mixture components in Fourier space.** Scalp plots for 0-Hz bin, real and imaginary values of 0.5-Hz bin. Components sorted by pathological class weight, also shown as colored text in top right of each component. Colormaps scaled per frequency bin. Note strong frontal components for mixture components associated with healthy class.

# 12 DISCUSSION

Deep-learning-based EEG decoding performance and interpretability can be further improved

- Deep networks we developed have competitive decoding performance
- Visualizations show networks learn well-known and surprising features
- Decoding performance gap between deep networks and feature-based decoding smaller than in other fields
- Cross-dataset, cross-electrode-configuration models may improve decoding performance
- Multimodal models can exploit more information and offer EEG $\rightarrow$ text and text $\rightarrow$ EEG synthesis
- In-context-learning may help decoding and interpretability

Finally, I conclude this thesis with my thoughts on the current state of EEG deep learning decoding and promising avenues for further work like cross-dataset decoding models, models that can process larger timescales of EEG signals, multimodal models and in-context learning.

## 12.1 STATE OF EEG DECODING USING OUR DEEP NETWORKS

Overall, our deep networks have shown good performance on a wide variety and settings of EEG brain-signal-decoding tasks, from classical movement-related trial-based decoding recording-based automatic pathology diagnosis. They can perform as well or better than feature-based baselines both on scalp and intracranial EEG. Here, fairly generic architectures like our deep ConvNet show robust performance across a wide variety of settings provided they are given enough training data.

Visualizations show these deep networks to learn well-known features like spectral amplitude, while also being capable of learning more complex features. Existing visualizations both reveal more complex waveforms than pure sinusoidal filters, as well as hierarchical features like a temporal increase in the amplitude of a learned frequency feature. Using invertible networks, we were even able to discover predictive features in less commonly used parts of the frequency spectrum.

On several datasets, the decoding performance gap between deeper networks and either smaller networks or even feature-based approaches it not as substantial as in other fields of machine learning like computer vision. Still, results show one advantage of deep networks, namely the possibility to use the same model across many tasks and settings, as the more generic network architectures can learn a wide variety of features suitable for different EEG decoding problems. Also, the results presented in this thesis show some promise to discover different learned EEG features through the use of deep learning.

## 12.2 FUTURE WORK

Using neural network architectures that can learn across datasets with different electrode configurations may help improve decoding performance. Here, transformer-based architectures [94] are a promising option, as they can be fed electrode coordinates as position encodings, potentially allowing to train them across datasets with different electrode configurations by simply supplying them the electrode coordinates of the current input. This could help to further increase the training data and thereby increase the EEG decoding performance.

Another architectural innovation for better decoding performance could be architectures that process larger time scales. Here, both transformed-based [9, 19, 27, 36, 67, 70, 101] and novel variants of convolutional architectures [24, 62] may be promising, as recent research has enabled them to process longer temporal sequences. This way, these architectures may for example look at an entire EEG recording at once to determine whether it is pathological. One challenge for this approach is that processing larger time windows instead of smaller ones decreases the training data again and more regularization may be needed.

Multimodal neural networks that can process the EEG signal as well as a textual description or other metadata could also improve decoding performance or used as interpretability tools. While models that get both text and signal as input could simply be used to improve decoding performance, models that go from textual description to EEG signal or vice versa [10, 82] may also help interpretability by textually summarizing a given EEG signal or visualizing a typical EEG signal corresponding to a specific textual report.

Finally, in-context learning is a method that might also lead to better EEG decoding performance by learning across different datasets and still exploiting the distribution of a specific dataset during inference. In-context-learning refers to trained networks that can learn to solve a novel task simply by being given input/output examples without further training [54, 57, 100]. Prominently observed in large language models, such behavior can also be explicitly trained for by training a model on entire labeled training datasets and unlabeled test datasets as input, optimizing to predict the correct

test labels [35, 57]. Given a sufficiently large EEG dataset, one may train such a model to process all the training data of a single subject to predict the test data of the same subject. Trained this way, it can learn robust features that work across subjects while still being able to exploit subject-specific features for prediction. One may also consider training on synthetic EEG data to have an unlimited number of datasets during training.

Additionally, combining in-context-learning with dataset condensation methods may help interpretability. Dataset condensation means to learn a smaller synthetic training dataset to replace the original training data [49, 97, 102, 103]. After training the in-context-learning model across many datasets, one could synthesize a small labeled training dataset that yields good performance on a given test dataset. Simply visualizing the examples in this synthesized training set may already reveal discriminative features, similar in spirit, but potentially more powerful than the class prototypes shown in Chapter 11.

## 12.3 CONCLUSION

Overall, EEG decoding using deep learning already works well, showing competitive decoding performance and revealing interesting learned features. Adopting more recent deep learning methods as the ones mentioned above may improve both aspects further.

**Open Questions**

- Can cross-dataset or long-time-scale learning lead to a substantial performance gain?
- Can multimodal or in-context learning help decoding performance and generate new insights into learned features?

# BIBLIOGRAPHY


[1] Reza Abiri, Soheil Borhani, Eric W Sellers, Yang Jiang, and Xiaopeng Zhao. "A comprehensive review of EEG-based brain–computer interface paradigms." In: *Journal of neural engineering* 16.1 (2019), p. 011001.

[2] Kai Keng Ang, Zheng Yang Chin, Haihong Zhang, and Cuntai Guan. "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)*. June 2008, pp. 2390–2397. DOI: 10.1109/IJCNN.2008.4634130. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4634130.

[3] A. Antoniades, L. Spyrou, C. C. Took, and S. Sanei. "Deep learning for epileptic intracranial EEG data." In: *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*. Sept. 2016, pp. 1–6. DOI: 10.1109/MLSP.2016.7738824.

[4] Lynton Ardizzone, Radek Mackowiak, Carsten Rother, and Ullrich Köthe. "Training Normalizing Flows with the Information Bottleneck for Competitive Generative Classification." In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin. 2020. URL: https://proceedings.neurips.cc/paper/2020/hash/593906af0d138e69f49d251d3e7cbed0-Abstract.html.

[5] Tonio Ball, Evariste Demandt, Isabella Mutschler, Eva Neitzel, Carsten Mehring, Klaus Vogt, Ad Aertsen, and Andreas Schulze-Bonhage. "Movement related activity in the high gamma range of the human EEG." In: *NeuroImage* 41.2 (June 2008), pp. 302–310. ISSN: 1053-8119. DOI: 10.1016/j.neuroimage.2008.02.032. URL: http://www.sciencedirect.com/science/article/pii/S1053811908001717 (visited on 07/15/2015).

[6] Pouya Bashivan, Irina Rish, Mohammed Yeasin, and Noel Codella. "Learning Representations from EEG with Deep Recurrent-Convolutional Neural Networks." In: *arXiv:1511.06448 [cs]*. arXiv: 1511.06448. 2016. URL: http://arxiv.org/abs/1511.06448 (visited on 12/20/2016).

[7] Joos Behncke, Robin T Schirrmeister, Wolfram Burgard, and Tonio Ball. "The signature of robot action success in EEG signals of a human observer: Decoding

and visualization using deep convolutional neural networks." In: *2018 6th international conference on brain-computer interface (BCI)*. IEEE. 2018, pp. 1–6.

[8] Joos Behncke, Robin Tibor Schirrmeister, Martin Volker, Jiri Hammer, Petr Marusic, Andreas Schulze-Bonhage, Wolfram Burgard, and Tonio Ball. "Cross-paradigm pretraining of convolutional networks improves intracranial EEG decoding." In: *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE. 2018, pp. 1046–1053.

[9] Iz Beltagy, Matthew E. Peters, and Arman Cohan. "Longformer: The Long-Document Transformer." In: *CoRR* abs/2004.05150 (2020). arXiv: 2004.05150. URL: https://arxiv.org/abs/2004.05150.

[10] Siddharth Biswal, Cao Xiao, M. Brandon Westover, and Jimeng Sun. "EEGtoText: Learning to Write Medical Reports from EEG Recordings." In: *Proceedings of the 4th Machine Learning for Healthcare Conference*. Ed. by Finale Doshi-Velez, Jim Fackler, Ken Jung, David Kale, Rajesh Ranganath, Byron Wallace, and Jenna Wiens. Vol. 106. Proceedings of Machine Learning Research. PMLR, 2019, pp. 513–531. URL: https://proceedings.mlr.press/v106/biswal19a.html.

[11] B. Blankertz, R. Tomioka, S. Lemm, M. Kawanabe, and K.-R. Muller. "Optimizing Spatial filters for Robust EEG Single-Trial Analysis." In: *IEEE Signal Processing Magazine* 25.1 (2008), pp. 41–56. ISSN: 1053-5888. DOI: 10.1109/MSP.2008.4408441. URL: http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=4408441.

[12] C. Brunner, R. Leeb, G. Müller-Putz, A. Schlögl, and G. Pfurtscheller. "BCI Competition 2008–Graz data set A." In: *Institute for Knowledge Discovery (Laboratory of Brain-Computer Interfaces), Graz University of Technology* (2008), pp. 136–142. URL: http://www.bbci.de/competition/iv/desc_2a.pdf (visited on 01/09/2017).

[13] Felix Burget, Lukas Dominique Josef Fiederer, Daniel Kuhner, Martin Völker, Johannes Aldinger, Robin Tibor Schirrmeister, Chau Do, Joschka Boedecker, Bernhard Nebel, Tonio Ball, et al. "Acting thoughts: Towards a mobile robotic service assistant for users with limited communication skills." In: *2017 European Conference on Mobile Robots (ECMR)*. IEEE. 2017, pp. 1–6.

[14] Hubert Cecotti and Axel Graser. "Convolutional Neural Networks for P300 Detection with Application to Brain-Computer Interfaces." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.3 (Mar. 2011), pp. 433–445. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2010.125. URL: http://dx.doi.org/10.1109/TPAMI.2010.125 (visited on 12/20/2016).

[15] Isha R Chavva, Anna L Crawford, Mercy H Mazurek, Matthew M Yuen, Anjali M Prabhat, Sam Payabvash, Gordon Sze, Guido J Falcone, Charles C Matouk, Adam de Havenon, et al. "Deep learning applications for acute stroke management." In: *Annals of Neurology* 92.4 (2022), pp. 574–587.

[16] Zheng Yang Chin, Kai Keng Ang, Chuanchu Wang, Cuntai Guan, and Haihong Zhang. "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*. Sept. 2009, pp. 571–574. DOI: 10.1109/IEMBS.2009.5332383. URL: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5332383.

[17] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)." In: *ArXiv e-prints*. Vol. 1511. 2016, arXiv:1511.07289. URL: http://adsabs.harvard.edu/abs/2015arXiv151107289C (visited on 12/21/2016).

[18] N. E. Crone, D. L. Miglioretti, B. Gordon, and R. P. Lesser. "Functional mapping of human sensorimotor cortex with electrocorticographic spectral analysis. II. Event-related synchronization in the gamma band." In: *Brain* 121.12 (Dec. 1998), pp. 2301–2315. ISSN: 0006-8950. DOI: 10.1093/brain/121.12.2301. URL: https://academic.oup.com/brain/article/121/12/2301/371496/Functional-mapping-of-human-sensorimotor-cortex (visited on 01/17/2017).

[19] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. "FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness." In: *NeurIPS*. 2022. URL: http://papers.nips.cc/paper_files/paper/2022/hash/67d57c32e20fd0a7a302cb81d36e40d5-Abstract-Conference.html.

[20] F. Darvas, R. Scherer, J. G. Ojemann, R. P. Rao, K. J. Miller, and L. B. Sorensen. "High gamma mapping using EEG." In: *NeuroImage* 49.1 (Jan. 2010), pp. 930–938. ISSN: 1053-8119. DOI: 10.1016/j.neuroimage.2009.08.041. URL: http://www.sciencedirect.com/science/article/pii/S1053811909009513 (visited on 01/10/2017).

[21] S. Lopez de Diego. "Automated interpretation of abnormal adult electroencephalography." MA thesis. Temple University, 2017.

[22] Laurent Dinh, David Krueger, and Yoshua Bengio. "NICE: Non-linear Independent Components Estimation." In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015. URL: http://arxiv.org/abs/1410.8516.

[23] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. "Density estimation using Real NVP." In: *CoRR* abs/1605.08803 (2016). arXiv: 1605.08803. URL: http://arxiv.org/abs/1605.08803.

[24] Daniel Y. Fu, Elliot L. Epstein, Eric Nguyen, Armin W. Thomas, Michael Zhang, Tri Dao, Atri Rudra, and Christopher Ré. "Simple Hardware-Efficient Long Convolutions for Sequence Modeling." In: *CoRR* abs/2302.06646 (2023). DOI: 10.48550/arXiv.2302.06646. arXiv: 2302.06646. URL: https://doi.org/10.48550/arXiv.2302.06646.

[25] Lukas AW Gemein, Robin T Schirrmeister, Patryk Chrabąszcz, Daniel Wilson, Joschka Boedecker, Andreas Schulze-Bonhage, Frank Hutter, and Tonio Ball. "Machine-learning-based diagnostics of EEG pathology." In: *NeuroImage* 220 (2020), p. 117021.

[26] A. Giusti, D. C. Cireşan, J. Masci, L. M. Gambardella, and J. Schmidhuber. "Fast image scanning with deep max-pooling convolutional neural networks." In: *2013 IEEE International Conference on Image Processing*. Sept. 2013, pp. 4034–4038. DOI: 10.1109/ICIP.2013.6738831.

[27] Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. "LongT5: Efficient Text-To-Text Transformer for Long Sequences." In: *Findings of the Association for Computational Linguistics: NAACL 2022*. 2022.

[28] Mehdi Hajinoroozi, Zijing Mao, Tzyy-Ping Jung, Chin-Teng Lin, and Yufei Huang. "EEG-based prediction of driver's cognitive performance by deep convolutional neural network." In: *Signal Processing: Image Communication* 47 (Sept. 2016), pp. 549–555. ISSN: 0923-5965. DOI: 10.1016/j.image.2016.05.018. URL: http://www.sciencedirect.com/science/article/pii/S0923596516300832 (visited on 12/20/2016).

[29] Jiří Hammer, Tobias Pistohl, Jörg Fischer, Pavel Kršek, Martin Tomášek, Petr Marusič, Andreas Schulze-Bonhage, Ad Aertsen, and Tonio Ball. "Predominance of Movement Speed Over Direction in Neuronal Population Signals of Motor Cortex: Intracranial EEG Data and A Simple Explanatory Model." In: *Cerebral Cortex (New York, NY)* 26.6 (June 2016), pp. 2863–2881. ISSN: 1047-3211. DOI: 10.1093/cercor/bhw033. URL: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4869816/ (visited on 01/11/2017).

[30] Kay Gregor Hartmann, Robin Tibor Schirrmeister, and Tonio Ball. "Hierarchical internal representation of spectral features in deep convolutional networks trained for EEG decoding." In: *2018 6th International Conference on Brain-Computer Interface (BCI)*. IEEE. 2018, pp. 1–6.

[31] Stefan Haufe, Frank Meinecke, Kai Görgen, Sven Dähne, John-Dylan Haynes, Benjamin Blankertz, and Felix Bießmann. "On the interpretation of weight vectors of linear models in multivariate neuroimaging." In: *NeuroImage* 87 (Feb. 2014), pp. 96–110. ISSN: 1053-8119. DOI: 10.1016/j.neuroimage.2013.

10.067. URL: http://www.sciencedirect.com/science/article/pii/S1053811913010914 (visited on 08/07/2015).

[32] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep Residual Learning for Image Recognition." In: *arXiv:1512.03385 [cs]* (Dec. 2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385 (visited on 05/11/2016).

[33] Felix A Heilmeyer, Robin T Schirrmeister, Lukas DJ Fiederer, Martin Volker, Joos Behncke, and Tonio Ball. "A large-scale evaluation framework for EEG deep learning architectures." In: *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE. 2018, pp. 1039–1045.

[34] Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. "Flow++: Improving Flow-Based Generative Models with Variational Dequantization and Architecture Design." In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, 2019, pp. 2722–2730. URL: http://proceedings.mlr.press/v97/ho19a.html.

[35] Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. *TabPFN: A Transformer That Solves Small Tabular Classification Problems in a Second*. 2022. DOI: 10.48550/ARXIV.2207.01848. URL: https://arxiv.org/abs/2207.01848.

[36] Delesley Stuart Hutchins, Imanol Schlag, Yuhuai Wu, Ethan S Dyer, and Behnam Neyshabur. "Block-Recurrent Transformers." In: *NeurIPS*. 2022. URL: https://arxiv.org/abs/2203.07852.

[37] Pavel Izmailov, Polina Kirichenko, Marc Finzi, and Andrew Gordon Wilson. "Semi-Supervised Learning with Normalizing Flows." In: *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*. Vol. 119. Proceedings of Machine Learning Research. PMLR, 2020, pp. 4615–4630. URL: http://proceedings.mlr.press/v119/izmailov20a.html.

[38] Pieter-Jan Kindermans, Kristof T. Schütt, Maximilian Alber, Klaus-Robert Müller, and Sven Dähne. "PatternNet and PatternLRP - Improving the interpretability of neural networks." In: *CoRR* abs/1705.05598 (2017). URL: http://arxiv.org/abs/1705.05598.

[39] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization." In: *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. 2014. URL: http://arxiv.org/abs/1412.6980 (visited on 01/09/2017).

[40] Diederik P. Kingma and Prafulla Dhariwal. "Glow: Generative Flow with Invertible 1x1 Convolutions." In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. Ed. by Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett. 2018, pp. 10236–10245. URL: https://proceedings.neurips.cc/paper/2018/hash/d139db6a236200b21cc7f752979132d0-Abstract.html.

[41] Zoltan J. Koles, Michael S. Lazar, and Steven Z. Zhou. "Spatial patterns underlying population differences in the background EEG." en. In: *Brain Topography* 2.4 (June 1990), pp. 275–284. ISSN: 0896-0267, 1573-6792. DOI: 10.1007/BF01129656. URL: http://link.springer.com/article/10.1007/BF01129656 (visited on 01/09/2017).

[42] Vernon J. Lawhern, Amelia J. Solon, Nicholas R. Waytowich, Stephen M. Gordon, Chou P. Hung, and Brent J. Lance. "EEGNet: A Compact Convolutional Network for EEG-based Brain-Computer Interfaces." In: *arXiv:1611.08024 [cs, q-bio, stat]* (Nov. 2016). arXiv: 1611.08024. URL: http://arxiv.org/abs/1611.08024 (visited on 12/20/2016).

[43] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." en. In: *Nature* 521.7553 (May 2015), pp. 436–444. ISSN: 0028-0836. DOI: 10.1038/nature14539. URL: http://www.nature.com/nature/journal/v521/n7553/full/nature14539.html (visited on 05/11/2016).

[44] R Leeb, C Brunner, GR Müller-Putz, A Schlögl, and G Pfurtscheller. "BCI Competition 2008–Graz data set B." In: *Graz University of Technology, Austria* (2008).

[45] X. Li and X. Wu. "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)*. Apr. 2015, pp. 4520–4524. DOI: 10.1109/ICASSP.2015.7178826.

[46] J. Liang, R. Lu, C. Zhang, and F. Wang. "Predicting Seizures from Electroencephalography Recordings: A Knowledge Transfer Strategy." In: *2016 IEEE International Conference on Healthcare Informatics (ICHI)*. Oct. 2016, pp. 184–191. DOI: 10.1109/ICHI.2016.27.

[47] Ilya Loshchilov and Frank Hutter. "SGDR: Stochastic Gradient Descent with Warm Restarts." In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL: https://openreview.net/forum?id=Skq89Scxx.

[48] Ilya Loshchilov and Frank Hutter. "Decoupled Weight Decay Regularization." In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL: https://openreview.net/forum?id=Bkg6RiCqY7.

[49] Dougal Maclaurin, David Duvenaud, and Ryan P. Adams. "Gradient-based Hyperparameter Optimization through Reversible Learning." In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*. Ed. by Francis R. Bach and David M. Blei. Vol. 37. JMLR Workshop and Conference Proceedings. JMLR.org, 2015, pp. 2113–2122. URL: http://proceedings.mlr.press/v37/maclaurin15.html.

[50] Ran Manor and Amir B. Geva. "Convolutional Neural Network for Multi-Category Rapid Serial Visual Presentation BCI." eng. In: *Frontiers in Computational Neuroscience* 9 (2015), p. 146. DOI: 10.3389/fncom.2015.00146.

[51] Ran Manor, Liran Mishali, and Amir B. Geva. "Multimodal Neural Network for Rapid Serial Visual Presentation Brain Computer Interface." In: *Frontiers in Computational Neuroscience* 10 (Dec. 2016). ISSN: 1662-5188. DOI: 10.3389/fncom.2016.00130. URL: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC5168930/ (visited on 02/03/2017).

[52] H. Mendoza, A. Klein, M. Feurer, J. Springenberg, and F. Hutter. "Towards Automatically-Tuned Neural Networks." In: *ICML 2016 AutoML Workshop*. June 2016.

[53] Risto Miikkulainen et al. "Evolving Deep Neural Networks." In: *arXiv:1703.00548 [cs]* (Mar. 2017). arXiv: 1703.00548. URL: http://arxiv.org/abs/1703.00548 (visited on 08/26/2017).

[54] Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. "Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?" In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Association for Computational Linguistics, 2022, pp. 11048–11064. URL: https://aclanthology.org/2022.emnlp-main.759.

[55] Mehdi Mirza and Simon Osindero. "Conditional generative adversarial nets." In: *arXiv preprint arXiv:1411.1784* (2014).

[56] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. "Methods for Interpreting and Understanding Deep Neural Networks." In: *CoRR* abs/1706.07979 (2017). URL: http://arxiv.org/abs/1706.07979.

[57] Samuel Müller, Noah Hollmann, Sebastian Pineda-Arango, Josif Grabocka, and Frank Hutter. "Transformers Can Do Bayesian Inference." In: *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL: https://openreview.net/forum?id=KSugKcbNf9.

[58] Fabian Nasse, Christian Thurau, and Gernot A. Fink. "Face detection using gpu-based convolutional neural networks." In: *International Conference on Computer Analysis of Images and Patterns*. Springer, 2009, pp. 83–90. URL: http://link.springer.com/chapter/10.1007/978-3-642-03767-2_10 (visited on 01/09/2017).

[59] A. Page, C. Shea, and T. Mohsenin. "Wearable seizure detection using convolutional neural networks with transfer learning." In: *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*. May 2016, pp. 1086–1089. DOI: 10.1109/ISCAS.2016.7527433.

[60] G Pfurtscheller. "Central beta rhythm during sensorimotor activities in man." In: *Electroencephalography and Clinical Neurophysiology* 51.3 (Mar. 1981), pp. 253–264. ISSN: 0013-4694. DOI: 10.1016/0013-4694(81)90139-5. URL: http://www.sciencedirect.com/science/article/pii/0013469481901395 (visited on 01/09/2017).

[61] G Pfurtscheller and A Aranibar. "Evaluation of event-related desynchronization (ERD) preceding and following voluntary self-paced movement." In: *Electroencephalography and Clinical Neurophysiology* 46.2 (Feb. 1979), pp. 138–146. ISSN: 0013-4694. DOI: 10.1016/0013-4694(79)90063-4. URL: http://www.sciencedirect.com/science/article/pii/0013469479900634 (visited on 01/09/2017).

[62] Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y. Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. "Hyena Hierarchy: Towards Larger Convolutional Language Models." In: *CoRR* abs/2302.10866 (2023). DOI: 10.48550/arXiv.2302.10866. arXiv: 2302.10866. URL: https://doi.org/10.48550/arXiv.2302.10866.

[63] F. Quandt, C. Reichert, H. Hinrichs, H. J. Heinze, R. T. Knight, and J. W. Rieger. "Single trial discrimination of individual finger movements on one hand: A combined MEG and EEG study." In: *NeuroImage* 59.4 (Feb. 2012), pp. 3316–3324. ISSN: 1053-8119. DOI: 10.1016/j.neuroimage.2011.11.053. URL: http://www.sciencedirect.com/science/article/pii/S1053811911013358 (visited on 01/17/2017).

[64] Maithra Raghu and Eric Schmidt. "A survey of deep learning for scientific discovery." In: *arXiv preprint arXiv:2003.11755* (2020).

[65] H. Ramoser, J. Muller-Gerking, and G. Pfurtscheller. "Optimal spatial filtering of single trial EEG during imagined hand movement." In: *IEEE Transactions on Rehabilitation Engineering* 8.4 (Dec. 2000), pp. 441–446. ISSN: 1063-6528. DOI: 10.1109/86.895946.

[66] V. Rau. "EEG Correlates of Inner Speech." In: *Bachelor's Thesis, University of Freiburg, DOI* (2015).

[67] Anirudh Ravula, Chris Alberti, Joshua Ainslie, Li Yang, Philip Minh Pham, Qifan Wang, Santiago Ontanon, Sumit Kumar Sanghai, Vaclav Cvicek, and Zach Fisher. "ETC: Encoding Long and Structured Inputs in Transformers." In: *2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*. 2020. URL: https://www.aclweb.org/anthology/2020.emnlp-main.19.pdf.

[68] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc Le, and Alex Kurakin. "Large-Scale Evolution of Image Classifiers." In: *arXiv:1703.01041 [cs]* (Mar. 2017). arXiv: 1703.01041. URL: http://arxiv.org/abs/1703.01041 (visited on 08/26/2017).

[69] Danilo Jimenez Rezende and Shakir Mohamed. "Variational Inference with Normalizing Flows." In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*. Ed. by Francis R. Bach and David M. Blei. Vol. 37. JMLR Workshop and Conference Proceedings. JMLR.org, 2015, pp. 1530–1538. URL: http://proceedings.mlr.press/v37/rezende15.html.

[70] Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. "Efficient Content-Based Sparse Attention with Routing Transformers." In: *Trans. Assoc. Comput. Linguistics* 9 (2021), pp. 53–68. DOI: 10.1162/tacl_a_00353. URL: https://doi.org/10.1162/tacl_a_00353.

[71] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak. "Convolutional, Long Short-Term Memory, fully connected Deep Neural Networks." In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Apr. 2015, pp. 4580–4584. DOI: 10.1109/ICASSP.2015.7178838.

[72] Haşim Sak, Andrew Senior, Kanishka Rao, and Françoise Beaufays. "Fast and Accurate Recurrent Neural Network Acoustic Models for Speech Recognition." In: *arXiv:1507.06947 [cs, stat]*. arXiv: 1507.06947. July 2015. URL: http://arxiv.org/abs/1507.06947 (visited on 12/21/2016).

[73] S. Sakhavi, C. Guan, and S. Yan. "Parallel convolutional-linear neural network for motor imagery classification." In: *Signal Processing Conference (EUSIPCO), 2015 23rd European*. Aug. 2015, pp. 2736–2740. DOI: 10.1109/EUSIPCO.2015.7362882.

[74] G. Schalk, D. J. McFarland, T. Hinterberger, N. Birbaumer, and J. R. Wolpaw. "BCI2000: a general-purpose brain-computer interface (BCI) system." In: *IEEE Transactions on Biomedical Engineering* 51.6 (June 2004), pp. 1034–1043. ISSN: 0018-9294. DOI: 10.1109/TBME.2004.827072.

[75] R. Schirrmeister, L. Gemein, K. Eggensperger, F. Hutter, and T. Ball. "Deep learning with convolutional neural networks for decoding and visualization of EEG pathology." In: *2017 IEEE Signal Processing in Medicine and Biology Symposium (SPMB)*. 2017, pp. 1–7. DOI: 10.1109/SPMB.2017.8257015.

[76] Robin Tibor Schirrmeister. "Convolutional Neural Networks for Movement Decoding from EEG Signals." MA thesis. Albert-Ludwigs-Universität Freiburg, 2015.

[77] Robin Tibor Schirrmeister, Jost Tobias Springenberg, Lukas Dominique Josef Fiederer, Martin Glasstetter, Katharina Eggensperger, Michael Tangermann, Frank Hutter, Wolfram Burgard, and Tonio Ball. "Deep learning with convolutional neural networks for EEG decoding and visualization." In: *Human Brain Mapping* (2017). ISSN: 1097-0193. DOI: 10.1002/hbm.23730. URL: http://dx.doi.org/10.1002/hbm.23730.

[78] Juergen Schmidhuber. "Deep Learning in Neural Networks: An Overview." In: *Neural Networks* 61 (Jan. 2015). arXiv: 1404.7828, pp. 85–117. ISSN: 08936080. DOI: 10.1016/j.neunet.2014.09.003. URL: http://arxiv.org/abs/1404.7828 (visited on 08/12/2015).

[79] Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, and Yann LeCun. "OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks." In: *arXiv:1312.6229 [cs]* (Dec. 2013). arXiv: 1312.6229. URL: http://arxiv.org/abs/1312.6229 (visited on 08/12/2016).

[80] Jared Shamwell, Hyungtae Lee, Heesung Kwon, Amar R. Marathe, Vernon Lawhern, and William Nothwang. "Single-trial EEG RSVP classification using convolutional neural networks." In: *SPIE Defense+ Security*. Ed. by Thomas George, Achyut K. Dutta, and M. Saif Islam. Vol. 9836. International Society for Optics and Photonics, May 2016. DOI: 10.1117/12.2224172. URL: http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.2224172 (visited on 02/14/2017).

[81] E. Shelhamer, J. Long, and T. Darrell. "Fully Convolutional Networks for Semantic Segmentation." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PP.99 (2016), pp. 1–1. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2016.2572683.

[82] Ana Maria Amaro de Sousa. "Learning to write medical reports from EEG data." In: (2022).

[83] Jost Tobias Springenberg. "Unsupervised and semi-supervised learning with categorical generative adversarial networks." In: *arXiv preprint arXiv:1511.06390* (2015).

[84] Sebastian Stober. "Learning Discriminative Features from Electroencephalography Recordings by Encoding Similarity Constraints." In: *Bernstein Conference 2016*. 2016. DOI: 10.12751/nncn.bc2016.0223.

[85] Sebastian Stober, Daniel J. Cameron, and Jessica A. Grahn. "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. Cambridge, MA, USA: MIT Press, 2014, pp. 1449–1457. URL: http://dl.acm.org/citation.cfm?id=2968826.2968988 (visited on 12/20/2016).

[86] Pascal Sturmfels, Scott Lundberg, and Su-In Lee. "Visualizing the Impact of Feature Attribution Baselines." In: *Distill* (2020). https://distill.pub/2020/attribution-baselines. DOI: 10.23915/distill.00022.

[87] Xuyun Sun, Cunle Qian, Zhongqin Chen, Zhaohui Wu, Benyan Luo, and Gang Pan. "Remembered or Forgotten?—An EEG-Based Computational Prediction Approach." In: *PLOS ONE* 11.12 (Dec. 2016), e0167497. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0167497. URL: http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0167497 (visited on 02/14/2017).

[88] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. "Rethinking the inception architecture for computer vision." In: *arXiv preprint arXiv:1512.00567* (2015). URL: http://arxiv.org/abs/1512.00567 (visited on 06/22/2016).

[89] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. "Intriguing properties of neural networks." In: *International Conference on Learning Representations*. 2014. URL: http://arxiv.org/abs/1312.6199.

[90] Yousef Rezaei Tabar and Ugur Halici. "A novel deep learning approach for classification of EEG motor imagery signals." en. In: *Journal of Neural Engineering* 14.1 (2017), p. 016003. ISSN: 1741-2552. DOI: 10.1088/1741-2560/14/1/016003. URL: http://stacks.iop.org/1741-2552/14/i=1/a=016003 (visited on 02/14/2017).

[91] Michael Tangermann et al. "Review of the BCI Competition IV." In: *Frontiers in Neuroscience* 6 (July 2012). ISSN: 1662-4548. DOI: 10.3389/fnins.2012.00055. URL: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3396284/ (visited on 08/20/2015).

[92] Lucas Theis, Aäron van den Oord, and Matthias Bethge. "A note on the evaluation of generative models." In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2016. URL: http://arxiv.org/abs/1511.01844.

[93] Pierre Thodoroff, Joelle Pineau, and Andrew Lim. "Learning Robust Features using Deep Learning for Automatic Seizure Detection." In: *JMLR Workshop and Conference Proceedings*. Vol. 56. 2016. URL: http://www.jmlr.org/proceedings/papers/v56/Thodoroff16.pdf (visited on 02/14/2017).

[94] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. "Attention is All you Need." In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett. 2017, pp. 5998–6008. URL: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[95] Martin Volker, Jiri Hammer, Robin T Schirrmeister, Joos Behncke, Lukas DJ Fiederer, Andreas Schulze-Bonhage, Petr Marusic, Wolfram Burgard, and Tonio Ball. "Intracranial error detection via deep learning." In: *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE. 2018, pp. 568–575.

[96] Martin Völker, Robin T Schirrmeister, Lukas DJ Fiederer, Wolfram Burgard, and Tonio Ball. "Deep transfer learning for error decoding from non-invasive EEG." In: *2018 6th International Conference on Brain-Computer Interface (BCI)*. IEEE. 2018, pp. 1–6.

[97] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A. Efros. "Dataset Distillation." In: *CoRR* abs/1811.10959 (2018). arXiv: 1811.10959. URL: http://arxiv.org/abs/1811.10959.

[98] X. Wang, C. A. Gkogkidis, R. T. Schirrmeister, F. A. Heilmeyer, M. Gierthmuehlen, F. Kohler, M. Schuettler, T. Stieglitz, and T. Ball. "Deep Learning for micro-Electrocorticographic (µECoG) Data." In: *2018 IEEE-EMBS Conference on Biomedical Engineering and Sciences (IECBES)*. 2018, pp. 63–68. DOI: 10.1109/IECBES.2018.8626607.

[99] Zuoguan Wang, Siwei Lyu, Gerwin Schalk, and Qiang Ji. "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. Beijing, China: AAAI Press, 2013, pp. 1785–1791. ISBN: 978-1-57735-633-2. URL: http://dl.acm.org/citation.cfm?id=2540128.2540384 (visited on 01/16/2017).

[100] Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. "An Explanation of In-context Learning as Implicit Bayesian Inference." In: *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL: https://openreview.net/forum?id=RdJVFCHjUMI.

[101] Manzil Zaheer et al. "Big Bird: Transformers for Longer Sequences." In: 2020. URL: https://proceedings.neurips.cc/paper/2020/file/c8512d142a2d849725f31a9a7a361ab9-Paper.pdf.

[102] Bo Zhao and Hakan Bilen. "Dataset Condensation with Differentiable Siamese Augmentation." In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 12674–12685. URL: http://proceedings.mlr.press/v139/zhao21a.html.

[103] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. "Dataset Condensation with Gradient Matching." In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL: https://openreview.net/forum?id=mSAKhLYLSsl.

[104] Barret Zoph and Quoc V. Le. "Neural Architecture Search with Reinforcement Learning." In: *arXiv:1611.01578 [cs]* (Nov. 2016). arXiv: 1611.01578. URL: http://arxiv.org/abs/1611.01578 (visited on 08/26/2017).

[105] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. "Learning Transferable Architectures for Scalable Image Recognition." In: *arXiv:1707.07012 [cs]* (July 2017). arXiv: 1707.07012. URL: http://arxiv.org/abs/1707.07012 (visited on 08/26/2017).