1 INTRODUCTION

TODO: textbox

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) [7], 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 [9, 21]. 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 [18].

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 [13, 24]. 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 maximally 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

TODO: textbox

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. [23].

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 **??**) on widely researched movement-related decoding tasks. TODO: Check if ref to section works

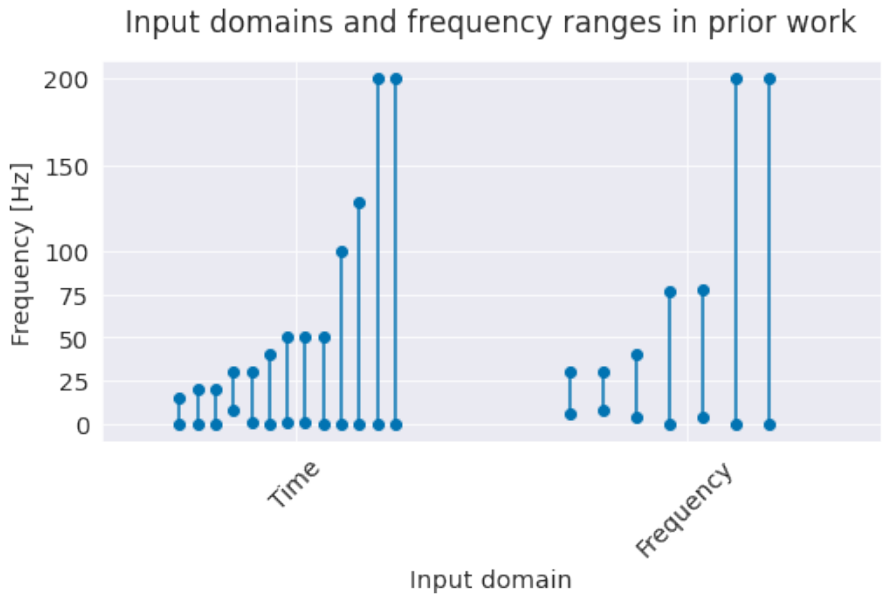

Figure 2.1: **Input domains and frequency ranges in prior work.** Grey lines represent frequency ranges of individual studies. Note that many studies only include frequencies below 50 Hz, some use very restricted ranges (alpha/beta band)

2.2 INPUT DOMAINS AND FREQUENCY RANGES

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 [23].

2.3 NETWORK ARCHITECTURES

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 evalu-

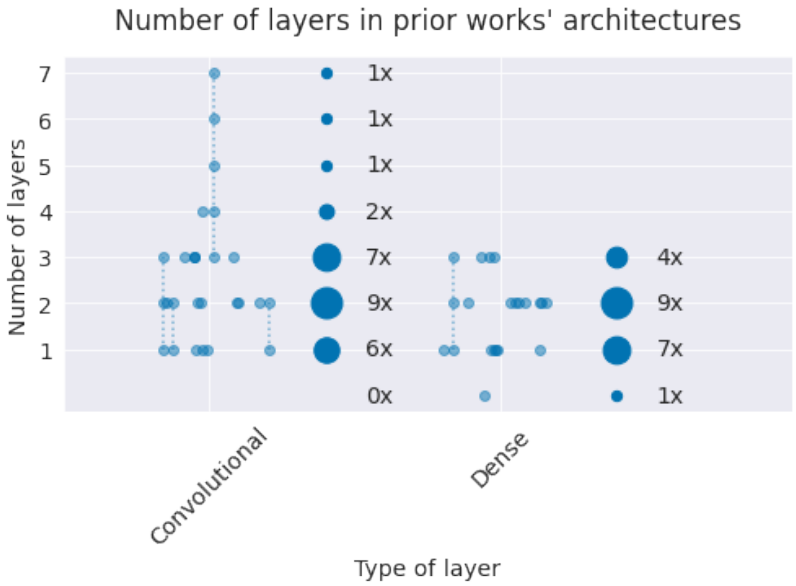

Figure 2.2: **Number of layers in prior work.** Small grey markers represent individual architectures. Dashed lines indicate different number of layers investigated in a single study (e.g., a single study investigated 3-7 convolutional layers). Larger grey markers indicate sum of occurences of that layer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

ated 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 [27]. 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 **??**. Later, we also developed more sophisticated visualization methods and applied them both to pathology decoding, see **??** and **??**.

TODO: open question textbox

STUDY	DESIGN CHOICES	TRAINING STRATEGIES
[12]	Kernel sizes	
[28]		Different time windows
[29]	Addition of six-layer stacked autoencoder on ConvNet features Kernel sizes	
[14]		Different subdivisions of frequency range Different lengths of time crops Transfer learning with auxiliary non-epilepsy datasets
[10]	Replacement of convolutional layers by restricted Boltzmann machines with slightly varied network architecture}	
[3]	1 or 2 convolutional layers	
[17]		Cross-subject supervised training, within-subject finetuning of fully connected layers
[4]	Number of convolutional layers Temporal processing of ConvNet output by max pooling, temporal convolution, LSTM or temporal convolution + LSTM	
[26]	Kernel sizes	Pretraining first layer as convolutional autoencoder with different constraints
[20]	Combination ConvNet and MLP (trained on different features) vs. only ConvNet vs. only MLP	
[27]	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
[32]	Partially supervised CSA	
[6]	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
[28]	Weights (spatial)	Largest weights found over prefrontal and temporal cortex.
[16]	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.
[29]	Weights (spatial + frequential)	Some weights represented difference of values of two electrodes on different sides of head.
[14]	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.
[31]	Input occlusion and effect on prediction accuracy	Allowed to locate areas critical for seizure.
[25]	Weights (spatial)	Some filter weights had expected topographic distributions for P300, others filters had large weights on areas not traditionally associated with P300.
[4]	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.
[26]	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.
[15]	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).
[6]	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

TODO: textbox

In a prior master thesis [22], 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, 8]) 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 [30]. 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 [5, 11, 19]. To do that, CSP 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:

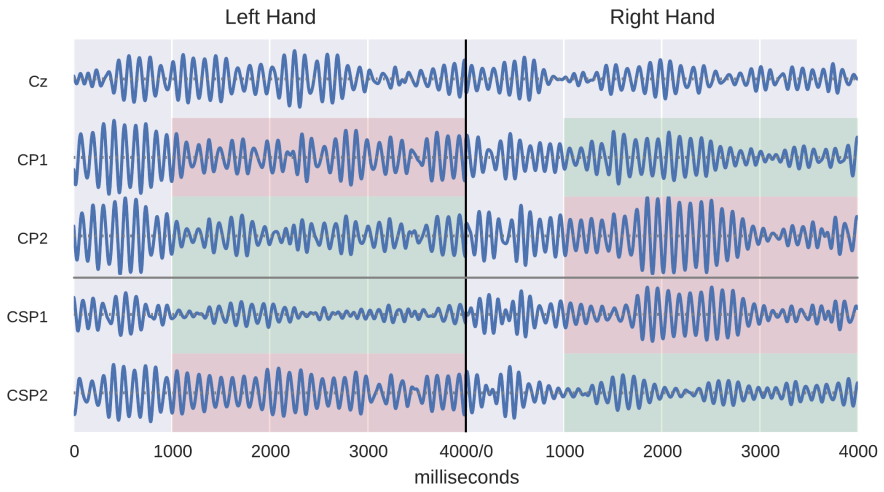

Figure 3.1: **Common Spatial Patterns example.** Top parts show EEG signals for three electrodes during a left hand and a right hand movement. Bottom parts show spatially filtered signals of two CSP filters. Green parts have lower variance and red parts have higher variance. Note that this difference is strongly amplified after CSP filtering. Figure from prior master thesis [22].

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

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, 8]. 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

The first neural network architecture was developed by us in a prior master thesis [22] 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:

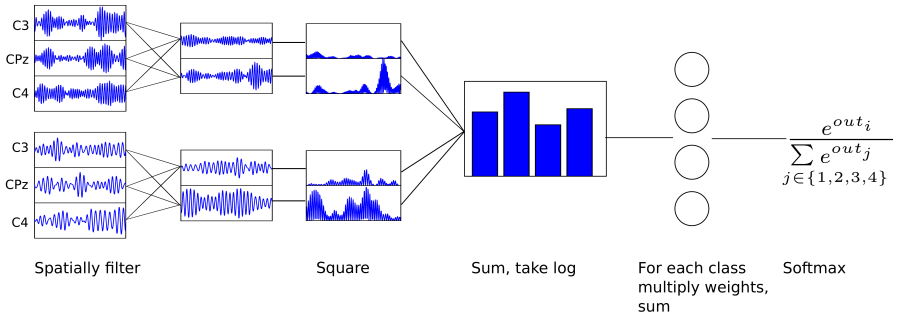

Figure 3.2: **Filter bank network architecture overview.** Input signals were bandpass filtered to different frequency ranges. Signals are first transformed by learned spatial filters, then squared, summed and the log-transformed. The resulting features are transformed into class probabilities by a classification weights followed by the softmax function. Figure taken from a master thesis [22].

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.

TODO: textbox