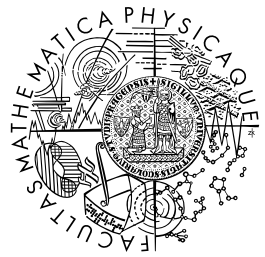


FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

MASTER THESIS

Michaela Vystrčilová

Prediction of velocity and speed of movement from human intracranial EEG recordings using deep neural networks

Department of Software and Computer Science Education

Supervisor of the master thesis: Mgr. Ján Antolík, PhD
Study programme: Computer Science
Study branch: IUI

Prague 2021

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In TELČ date 20.5.2021 .....

Author's signature

First, I would like to thank my supervisor Mgr. Ján Antolík, Ph.D. for introducing me to the intriguing field of computational neuroscience. His supportive attitude and advice helped me way beyond this thesis. In a similar manner I would like to thank Mgr. Jiří Hammer, Ph.D. who offered me not only this project but importantly also his time and knowledge.

I also want to express my gratitude to Mr. Robin Tibor Schirrmeister, soon to be Ph.D. and Prof. Tonio Ball for their welcoming consultations regarding the Deep4Net and brain signal processing in general. Their guidance was invaluable.

Further I would like to manifest my appreciation for at least some of the people who, more or less directly, contributed to me starting and also finishing this thesis. For encouraging me not to content myself with something I would not enjoy doing, for standing me in times of quarantine and for all the emotional support and food, thank you, Vít Kabele, Jana, Jiří, Jana and Lenka Vystrčilovi and Anna Bláhová.

Last, but not least, I would like to thank my cat for sitting on my lap long enough for me to be able to finish the thesis.

Title: Prediction of velocity and speed of movement from human intracranial EEG recordings using deep neural networks

Author: Michaela Vystrčilová

Department: Department of Software and Computer Science Education

Supervisor: Mgr. Ján Antolík, PhD, Department of Software and Computer Science Education

Abstract: Our brain controls the processes of the body including movement. In this thesis, we try to understand how the information about hand movement is encoded into the brain's electrical activity and how this activity can be used to predict the velocity and absolute velocity of hand movements. Using a well-established deep neural network architecture for EEG decoding - the Deep4Net - we predict hand movement velocity and absolute velocity from intracranial EEG signals. While reaching the expected performance level, we determine the influence of different frequency bands on the network's prediction. We find that modulations in the high-gamma frequency band are less informative than expected based on previous studies. We also identify two architectural modifications which lead to higher performances. 1. the removal of max-pooling layers in the architecture leads to significantly higher correlations. 2. the non-uniform receptive field of the network is a potential drawback making the network biased towards less relevant information.

Keywords: motor decoding, intracranial EEG (iEEG), deep learning, convolutional neural networks

Contents

Chapter 1

Introduction

Historical context

While mankind has been fascinated by the human mind for a long time, only the last 70 years of rapid technological advances finally allowed us to peer deep into the human brain. New recording techniques, theoretical models, and detailed simulations facilitated by the increase in computational power have opened new horizons in the field of neuroscience. However, to this day, a lot remains unclear about how the activity of billions of neural cells interconnected through a dense web of dynamic circuits translates into complex processes such as perception, movement, reasoning or learning.

One of the recent sub-fields of neuroscience, namely *computational neuroscience*, combines neuroscience with mathematics and computer science to formulate theoretical principles governing the nervous system. The earliest theoretical discoveries in neuroscience were models of single neurons (Brunel et al., 2008; Hodgkin et al., 1952). Modelling activity of single neurons, however, cannot capture the brain in its whole complexity. Because the billions of neurons and other neural cells are constantly communicating with each other, their coordinated activity is another key to unlocking the brain's mysteries. Methods capturing and interpreting such activity are therefore needed. In 1924, Hans Berger was the first to record human brain activity through electroencephalography (EEG). Since then, many other recording techniques for measuring aggregate signals from large populations of neurons emerged, such as magnetoencephalography (MEG), functional magnetic resonance imaging (fMRI), positron emission tomography (PET), magnetic resonance spectroscopy (MRS), etc.

There is a long path from recording brain activity to understanding how it translates into processes the brain governs in humans. One approach to formulating a relationship between brain signal and a certain process such as move-

ment or sleep is called *brain signal decoding* or also *neural decoding*. It studies what information is available in the electrical activity of neurons or networks of neurons by mapping patterns in the electrical activity onto external processes such as movement or sleep. To create such a mapping, first, the brain signal is recorded simultaneously with a variable describing the process. Then a model predicting the variable based on the brain signal is built and fitted to the recorded data.

These models allow for a deeper understanding of the information contained in brain activity and direct clinical applications. They are being used for predicting epileptic seizures (Hassan et al., 2016), Alzheimer disease diagnosis (Ahmadlou et al., 2011), classifying sleep stages (Şen et al., 2014) or in Brain-computer interfacing (BCIs) (Schalk et al., 2011; Wolpaw et al., 1991).

Deep artificial neural networks (DNNs) are being increasingly utilized as such models (Roy et al., 2019). DNNs elevated what is considered state-of-the-art in various fields, most prominently computer vision (Brock et al., 2021) and natural language processing (Rae et al., 2019). Their ability to process complex data in an end-to-end manner and good prediction performance makes them suitable for decoding from brain signals.

Motivation

Utilizing recent advances of deep learning (DL) in movement decoding from EEG has brought considerable progress in this field. In multiple cases, deep neural networks have been shown to be more effective when decoding from EEG and intracranial EEG (iEEG) than standard machine learning methods (Lawhern et al., 2018; Mousavi et al., 2019; Zhang et al., 2019). Nevertheless, there is still room for considerable improvement in the decoding accuracy, interpretability and applicability to online decoding (Roy et al., 2019). These issues are even more substantial in movement decoding from intracranial EEG. Because it is more difficult to acquire iEEG data, it is less researched but very promising because of the higher signal-to-noise ratio of iEEG signal compared to EEG (Volkova et al., 2019).

To address these issues, Hammer et al., 2021 employed the Deep4Net - a convolutional neural network introduced in Schirrmeister et al., 2017 - to perform iEEG movement decoding, namely to decode hand velocity and absolute velocity (speed) from intracranial EEG. The Deep4Net was chosen because in previous studies (Hartmann et al., 2018; Schirrmeister et al., 2017) it has proven successful in movement-related classification tasks, reaching comparable accuracy with state-of-the-art methods without the necessity of manual feature extraction. When using it, Schirrmeister et al., 2017 also inspected which frequency

bands are essential for the network's decisions. Besides confirming the informativeness of the alpha (4 - 12 Hz) and beta (13 - 30 Hz) bands, it was the first time that the importance of modulations in the high-gamma (70 Hz and above) band was shown for movement decoding from non-invasive EEG. The fact that Deep4Net architecture can extract information from the high-gamma band suggests its suitability for iEEG decoding, where the high-frequency resolution is better than in non-invasive EEG (Light et al., 2010).

Hammer et al., 2021 found that the Deep4Net significantly outperforms multi-linear regression in velocity and absolute velocity decoding. Nevertheless, when visualizing which features it focuses on, the network has not shown the expected interest in the high-gamma frequency modulations. In the context of previous findings on the same dataset, where high-gamma was found significant, especially for absolute velocity decoding when using multi-linear regression (Hammer et al., 2016) and the fact that the Deep4Net architecture is able to extract information from the high-gamma frequency (Schirrmeister et al., 2017), the group briefly continued to explore this surprising finding.

Using a different visualization method, namely gradient visualization, they discovered a gradient peak in the high-gamma range at 83.33 Hz, showing possible interest of the network in information from the high-gamma frequency (personal communication). Multiple questions remain to be answered at this point: Is the gradient peak significant, or is it just an architectural artefact? If it is an architectural artefact and does not show the use of modulations in the high-gamma frequency band by the network, why did the network not use the high-gamma? Would other architectures be able to utilize high-gamma in this setting? Could we improve performance by forcing the network to use high-gamma? Is high-gamma indeed informative for decoding (absolute) velocity of hand movement? If the peak is not an architectural artefact but reflects the use of information from the high-gamma band by the network, why did the perturbation visualization method performed by Hammer et al., 2021 not show it?

Goals

The aim of this thesis is to address the above-outlined questions, which can be summarized in the two following goals:

1. Understanding which frequency bands are utilized by the Deep4Net for hand movement decoding with particular focus on the high-gamma band.

2. Identifying which modifications to DNN training/architecture improve the utilization of information across useful frequency bands.

Achieving these goals will contribute to the interpretability of deep neural networks used for iEEG movement decoding. By identifying informative frequency bands and optimizing the network architecture to use them effectively, we can also potentially improve the decoding performance of these networks, advancing their clinical application.

Chapter 2

Background and related work

This chapter gives a brief introduction of topics most relevant for this thesis to equip the reader with the necessary background. We start by shortly describing (intracranial) EEG signals, their standard feature extraction, and decoding techniques (Section 2.1). We further provide a concise introduction to deep neural networks (DNNs), paying particular attention to convolutional neural networks (CNNs) (Section 2.2). Lastly, we introduce the two most relevant papers (Hammer et al., 2021) and (Schirrmeister et al., 2017) in more detail, followed by brief descriptions of other DNN architectures used for movement decoding (Section 2.3).

2.1 Decoding from (i)EEG signals

Neural decoding refers to a neuroscience field concerned with reconstructing external stimuli from information that is already encoded in the brain. It consists of multiple stages: signal recording, spectral and spatial feature extraction, and the final classification or regression algorithm. This section describes the methods traditionally used in each of these stages, highlighting their advantages and disadvantages. While many things, such as sleep stages (Mousavi et al., 2019), epileptic seizures (Hassan et al., 2016) or emotions (Zheng et al., 2015) can be decoded from brain signals, we introduce mainly methods used for movement decoding because they are most relevant for this thesis.

2.1.1 Recording methods

An existing correlation between neural modulations and motor parameters for a wide range of motor tasks makes electrical brain activity suitable for movement decoding (Lebedev et al., 2005). Electroencephalography (EEG), electrocorticog-

raphy (ECoG), and stereotactic EEG (sEEG) are the most commonly used techniques of recording the electrical activity of the brain (Tam et al., 2019). We describe the basic properties of each of these recording methods. For a more detailed introduction to EEG, please refer to Schomer et al., 2017.

Electroencephalography (EEG)

Electroencephalography (EEG) records electrical brain activity of the cerebral cortex using multiple electrodes (also called channels), which are placed on the surface of the head. The recorded activity, originating as the postsynaptic potential of excitable neural tissue (Buzsaki et al., 2012), is attenuated through cerebrospinal fluid, the dura, and the skull on its way to the recording electrodes. Besides weakening the signal, these structures also act as low-pass filters limiting the useful frequency bands to below 100 Hz (Tam et al., 2019). Another limitation of EEG recordings is the interference of muscles located in the head region. Muscle activity has higher amplitudes than brain signals and can affect the recording quality (Scholg et al., 2002). It lowers the signal-to-noise ratio of EEG signals. EEG has also relatively poor spatial resolution (Buzsaki et al., 2012). Despite the aforementioned drawbacks, EEG is a widely used tool in brain signal recording. Its high temporal resolution, non-invasiveness, and relatively low cost motivate researchers to develop new quality control and artefact processing techniques (Lotte et al., 2018) to alleviate the disadvantages of non-invasive EEG.

Electrocorticography and stereotactic EEG

Electrocorticography (ECoG) and stereo-tactic electroencephalography (sEEG) are variations of EEG commonly referred to as intracranial EEG (iEEG). In ECoG, a grid with multiple recording electrodes (channels) is placed on the brain's surface instead of the surface of the head. For stereotactic EEG, the electrodes penetrate the skull and reach deeper inside the brain. Both are very invasive methods and cannot be performed for the sake of research only. Therefore, intracranial EEG studies are mainly conducted on patients with medication-resistant epilepsy. The electrodes are used to find the locus of seizures which is then surgically removed. Special measures have to be taken to remove channels displaying pathological symptoms when using ECoG signal for research. The quality of signals from non-epileptic brain tissue is, however, superior to standard EEG. It has a higher spatial resolution and better resolution in the high frequencies (Volkova et al., 2019). with medication-resistant epilepsy. The electrodes are used to find the locus of seizures which is then surgically removed. Special measures have to be taken to remove channels displaying pathological symptoms when using ECoG signal for research. The quality of signals from non-epileptic brain tis-

sue is, however, superior to standard EEG. It has a higher spatial resolution and better resolution in the high frequencies (Volkova et al., 2019).

2.1.2 Feature extraction

End-to-end learning has started to be used only recently, and manual feature extraction is common when designing movement decoders from EEG and iEEG. We will describe some widely used feature extraction methods. Often distinct methods are used for spatial and temporal information present in EEG and iEEG signals. Spatial information tells us where in or on the brain the signal was measured. It depends on the position of the electrode. Temporal information contains information about when the signal was recorded and what its value was at that moment.

Spatial feature extraction

Spatial feature extraction methods are used for signal denoising, rejection of pathological channels, and dimensionality reduction. Common average reference (CAR) is a widely used and straightforward denoising method. In CAR, the mean of all channels is subtracted from each channel (Liu et al., 2015). This reduces noise common to all channels but might introduce some channel-specific noise to some otherwise clean channels (Volkova et al., 2019). Multiple other methods have been proposed to address this issue. For example, auto-regressive models, which instead of subtracting the mean of the channels assign a weight to each channel to minimise the variance, and then subtract this weighted average from each channel (Lu et al., 2012). Bipolar reference and Laplacian reference (Li et al., 2018; Yao et al., 2019) are also ways to avoid proposing channel-specific noise by focusing only on the re-referencing of neighboring channels.

Noisy and epileptic channel rejection is often done after visual inspection by experts. Nevertheless, automatic mechanisms to select channels for removal have also been developed. A noisy channel can be identified based on an abnormal distribution, identification of excessive line noise, or the outliers it contains (Liu et al., 2015). If a channel is marked noisy, it is removed from the dataset. However, some studies propose the use of CAR with median subtraction instead of mean subtraction while keeping all channels. They claim that this way, it is possible to retain all information potentially valuable for decoding while also sufficiently reducing noise across all channels (Liu et al., 2015).

Recording from multiple electrodes creates highly dimensional data. Reducing their complexity is another opportunity for spatial feature extraction to help achieve better decoding results. Principal component analysis (PCA) (Pearson, 1901) is a commonly used method (Volkova et al., 2019). It transforms the data

into a new coordinate system achieving the largest possible variance along the axes. Optionally, one can then discard the axes with lower variance to achieve dimensionality reduction. Independent component analysis, another tool for dimensionality reduction, also transforms the data into a new coordinate system. However, unlike PCA, it selects a basis in which each vector is an independent component of the data (Michelmann et al., 2018).

Spectral feature extraction

Spectral feature extraction methods are used to extract task-related features from the time-domain and power spectra domain. Regarding the time-domain, the Low-pass filtered component (LFC), also called Local motor potential (LMP), which can be extracted using Savitzky-Golay filters (Pistohl et al., 2011), has been shown by multiple studies to hold information about movement time-course and kinematics (Ball et al., 2009; Pistohl et al., 2008; Schalk et al., 2007).

In the power spectrum domain, following are the traditionally defined frequency ranges: δ (0–4 Hz), θ (4–8 Hz), α (8–12 Hz), β (13–30 Hz), low γ (30–70 Hz) and in the case of ECoG high γ band (70 Hz and above) (Hammer et al., 2016). The modulations in the alpha, beta, and in the case of intracranial EEG also high-gamma bands are particularly informative for movement decoding (Ball et al., 2008; Gunduz et al., 2016). As previously stated, the high-gamma signal from EEG is often noisy and not suitable for decoding. However, it has been recently shown that neural networks are able to use information from the high-gamma band for movement classification (Schirrmeister et al., 2017).

Multiple methods have been proposed to extract features from the power spectrum domain. The signal can be either directly band-filtered in the desired range or converted into the frequency domain using both non-parametric and parametric methods. Non-parametric methods include Fourier analysis, multitaper methods, and wavelet transformations (Volkova et al., 2019).

Fourier analysis involves convolving the brain signal with a sinusoidal function of a fixed length. The analysis is performed on short windows to improve temporal specificity; nevertheless, its time-frequency resolution is relatively poor. Moreover, it is only useful in a limited frequency range, based on the window size, and the size also fixes the scale of the signal that is to be detected (Van Vugt et al., 2007). Both multi-tapering methods and wavelets were designed to ameliorate these issues.

Decomposing the signals into shifted and scaled versions of an oscillating waveform of a wavelet function allows the proportion between the temporal width and frequency bandwidth to stay the same for all frequencies. This contributes to a higher temporal resolution in higher frequencies. The good time-frequency trade-off of wavelets makes them suitable for analysing non-

stationary signals (Van Vugt et al., 2007).

Multi-taper methods were designed to reduce bleeding between frequencies, making them, too, well-suited for analysing signals of non-stationary processes. Tapers are functions having a value of one in the middle and then taper to zero at the edges. The multi in multi-taper implies that the signal is convolved with multiple such orthogonal tapers. These are then averaged, yielding a statistically consistent spectral estimator with a reduced variance of the spectral estimates and improved localisation in the frequency domain (Pistohl et al., 2011).

Auto-regressive spectral estimators (AR) are parametric spectral feature extractors. They have proven successful for modelling both EEG (Walter et al., 2012) and iEEG (Anderson et al., 2009). Even though ARs have an inherent capacity to model peaky spectra characteristic for bio-signal, their effectivity largely depends on parameterisation, which is not particularly straightforward (Anderson et al., 2009).

More recently proposed approaches, based on Riemann geometry, which use covariance matrices for feature learning and representation, have shown promising results (Delgado Saa, 2020). Temporal modulations can also be modelled probabilistically using hidden Markov's models or conditional random fields (Milstein et al., 2017; Wang et al., 2011). Importantly, also neural networks can be integrated into the movement decoding pipeline as feature extractors. For example auto-encoders are suited for dimensionality reduction (Wen et al., 2018). An auto-encoder consists of two parts. The encoder first compresses the data into a smaller representation. The decoder then receives the compressed data and tries to reconstruct them without information about their initial form. This forces the encoder to create a smaller representation, which contains as much information as possible. Convolutional neural networks are also particularly convenient for signal processing because many of the filters mentioned above can be implemented as convolutions.

2.1.3 Classification methods

Various movement related classification tasks such as decoding wrist flexion or extension (Edelman et al., 2016) movement of individual fingers (Hotson et al., 2016; Saa et al., 2016) or feet movement (Satow et al., 2003) from EEG and iEEG have been studied using a diverse population of classifiers. The classifiers include Support vector machines (SVM) (Hearst et al., 1998), Linear Discriminant analysis (LDA) (Mika et al., 1999), Quadratic discriminant analysis (QDA) (Friedman, 1989), their adaptive versions which have gained popularity in the last ten years (Lotte et al., 2018) and probabilistic methods such as conditional random fields or naive bayes classifiers (Saa et al., 2016; Wang et al., 2011).

2.1.4 Regression methods

Regression tasks attracted less attention from researchers than classification tasks (Volkova et al., 2019). Still, extensive research with many different kinds of tasks was conducted on this front as well. Movement decoding tasks which fall in the category of regression include finger position decoding (Pistohl et al., 2008), hand position decoding (Ball et al., 2009) or kinematic variables such as speed or velocity decoding of hand movements (Hammer et al., 2013, 2016, 2021; Lv et al., 2010; Ofner et al., 2012). Various algorithms have been employed to solve the above-listed tasks. Models based on linear regression combined with various feature extraction methods are very common (Hammer et al., 2013, 2016; Hotson et al., 2014; Ofner et al., 2012). More sophisticated models include Kalman filters (Lv et al., 2010) and their unscented versions (Luu et al., 2016). Neural networks, which we describe separately in Section 2.3 have also been proposed to solve movement-related regression tasks.

2.2 Artificial neural networks

In this section, we provide a quick overview of the fundamentals of deep learning (DL). We introduce basic principles of deep neural networks (DNNs) relevant for this thesis.

2.2.1 Convolutional neural networks

DNNs are statistical machine learning models which are somewhat based on the functioning of the brain. The networks are composed of interconnected neurons, also called units. A neuron in an artificial neural network is defined as a weighted sum of its inputs followed by an activation function which is usually non-linear. Multiple layers of interconnected neurons are stacked on top of each other in deep neural network architectures, thus the depth in the term. The neurons in the layers can be interconnected in various ways. Feed-forward networks are such that the connections do not form a cycle. The outputs of the preceding layer are always used as inputs of the following one. Networks in which the connections do form cycles are called Recurrent neural networks (RNNs). For more information on RNNs, please refer to Lipton, 2015.

In feed-forward networks, to which we limit the introduction, a simple approach of connecting neurons between two layers connects every pair of neurons with a weight creating a *fully connected* network. Consider layers A with a neurons and B with b neurons. The output of a neuron k from layer B is then defined

as:

$$y_k = f(\sum_{i=1}^{a} x_i * w_{i,k}) \tag{2.1}$$

where f is the activation function, x_i is the output of neuron a_i and $w_{i,k}$ is the weight between neuron a_i and b_k. An example of a simple three-layer fully connected neural network is displayed in Figure 2.1.

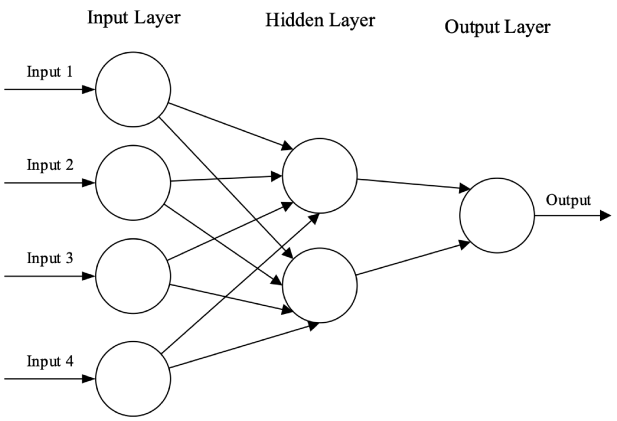


Figure 2.1 An example of a three-layer fully connected feed-forward network taken from OShea et al., 2015.

Besides assigning a separate weight to each pair of neurons, there are other possibilities of connecting neurons of consecutive layers. Instead of assigning one weight to each pair of neurons, we can create a smaller weight matrix, often called filter or kernel, which we reuse for multiple pairs of neurons. To do so, we perform convolution (i. e., a sliding dot product between the filter and a correspondingly sized part of the layer's input matrix). For visual explanation see Figure 2.2. More parameters need to be specified to arrive at a complete convolutional layer from the convolutional filter:

- W and H - the width and height of the kernel
- D the depth of the kernel. It specifies how many input channels the layer's input has. What channels can represent can be illustrated, for example, on images. Imagine a coloured image of size 32x32 pixels. Besides the width and height, the image also has a 3rd dimension specifying the RGB colour. This third dimension is the channels; in this case, there are three.

Therefore, the dimension of the input (the image) is 32x32x3 and the kernel sliding over it has to have an additional dimension to handle the third input dimension. This additional dimension is the depth.

- F - the number of output channels. It specifies the number of different filters of size $WxHxD$, which all slide over the input giving F output channels. Similarly to the RGB channels of the input described above, each of the different filters allows the network to create its own representation of some feature of the layer's input.

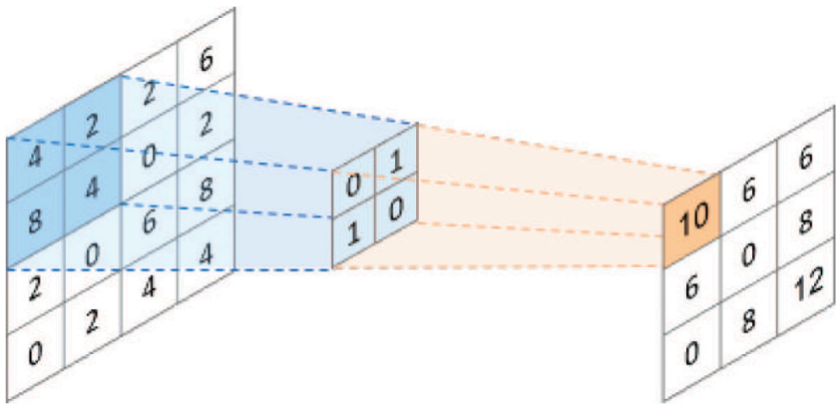
- *stride* - it specifies by how many input points the kernel is shifted when sliding over the input.

- *padding* - specifies if only valid inputs should be taken into account or if the input is to be padded when the kernel reaches outside of the original input. If no padding is involved, the size of the output decreases along the dimension where the kernel size of the filter is not 1.

Consider a 2D convolutional layer with input I and with D input channels, which is parameterised by a kernel K of total size $WxHxDxF$. The output of this layer is computed as:

$$(K * I)_{i,j,o} = f(\sum_{m,n,d} I_{i*s_1+m,j*s_2+n,d} K_{m,n,d,o}) \tag{2.2}$$

where f is the activation function, s_1 and s_2 are strides in the corresponding dimensions, d are the input channels and o are the output channels.



.

Figure 2.2 An example of a convolutional filter of width 2 and height 2 with weights 0, 1, 1, 0 sliding over an input of size 4x4 taken from Tsai et al., 2020

When a network contains one or multiple layers performing convolution, it is called a convolutional neural network (CNN). Convolutional layers are often supplemented with pooling layers such as max-pooling or average pooling, which aim to achieve shift-invariance in CNN architectures (Gu et al., 2018). Common activation functions used in CNNs are sigmoid, tanh, or ReLU (Nair et al., 2010). A more recent and popular activation is the ELU function we are using in this thesis (Clevert et al., 2016).

Training

Deep neural networks tend to have large numbers of parameters thus cannot be trained analytically. Instead, they are trained using fitting based methods. These methods are variations of the gradient descent back-propagation algorithm (Rumelhart et al., 1987). Some popular variations include the Stochastic gradient descent (Bottou, 2010), Adagrad (Duchi et al., 2011) or more recently ADAM (Kingma et al., 2017). To give the reader an idea of the fitting process, we describe it in a simplified manner. First, a small subset of the input (a *batch*) is randomly sampled from the training set. The batch is used as input into the network and is passed through the networks in a *forward pass* to get the network's output. The output and the *gold labels* (desired output based on the selected inputs) are given to the *loss function* which serves as an indicator of how far the model is from the correct output. Finally, the first derivatives of the loss function (the *gradient*) with respect to the parameters (the *weights*) of the model are subtracted from these parameters. This allows moving in the direction of the steepest descent in the loss function value. In theory, this decreases the value of the loss function in the next forward pass. To limit the size of the descent steps, we use the *learning rate* by which we multiply the gradient. The forward pass is repeated for all batches in the dataset. When all batches pass through the network, we can say one *epoch* was completed. The goal is to train a network for a number of epochs until it *converges* to a low value of the loss function.

A low loss value on the training set means the model predicts the training data well. Nevertheless, to see if the model can generalise over unseen data, we keep another portion of data - the validation set - out of training and only use it at the end of each epoch to estimate how it performs on unseen data.

Regularization

Overfitting - the adaptation to the training set distribution to the extent that the model does not generalise well over unseen data - is an unneglectable problem in deep neural networks, including CNNs. Multiple methods which effectively reduce this problem have been proposed. One option is to add a penalisation

term into the loss function, which penalises the model complexity. An example of such a regularization is l_p-norm regularization (Gu et al., 2018). Dropout layers (Hinton et al., 2012) also serve as regularization tools. By randomly dropping a portion of the connections between neurons, they prevent the network from becoming dependant on single neurons and force it to be able to make good predictions even when some information is absent. Another popular regularisation technique for CNNs is batch normalisation (Ioffe et al., 2015) which, besides regularisation, also reduces the internal covariance shift. In CNNs, this method now often substitutes dropout layers which are more suited for fully connected layers (Gu et al., 2018).

Dilated networks

Convolutional neural networks are widely used in computer vision, specifically in pattern recognition, where they substantially raised state-of-the-art (Brock et al., 2021; Krizhevsky et al., 2012), and their localised focus is well suited to image processing. Nevertheless, CNNs are increasingly used for dense predictions. Dense prediction in the sense of computer vision means assigning a label to each pixel of an image (Long et al., 2015). But this notion of one output per one input point can be extended to other fields such as machine translation (Kalchbrenner et al., 2016), speech synthesis (Angrick et al., 2018) or prediction from EEG (Schirrmeister et al., 2017) and iEEG (Hammer et al., 2021). The above-mentioned tasks were solved using CNNs where an additional parameter, called dilation, was introduced (Long et al., 2015). Dilation creates gaps between the convolutional filter elements and enlarges the CNN's receptive field, allowing it to cover more relevant information. This way, the networks are more suitable for sequence processing (Gu et al., 2018). For a visual explanation of dilation, see Figure 2.3.

2.2.2 CNN interpretability

Poor interpretability of deep neural networks, including CNN models, has been identified as a major challenge limiting their usefulness across fields. Especially in safety-critical fields such as autonomous driving or medical decision support, is it crucial to understand how DNNs arrive at their decisions. Furthermore, in the case of decoding from EEG and iEEG signals, understanding how networks, which are able to translate these signals into, for example, velocity or speed, arrive at their decision would lead to a better understanding of the brain's functionality. It could also help identify ways in which these networks can be optimised to yield better predictions. Because the strength of DNNs often lies in large datasets, in a field where data is less available, such optimisations are

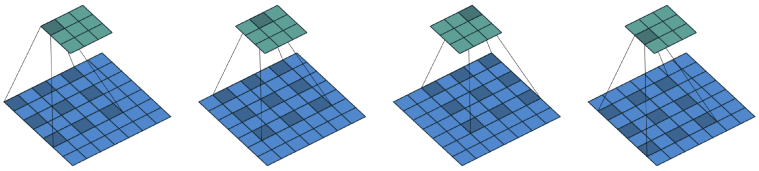


Figure 2.3 An illustration of a dilated convolution with kernel size (3, 3) and dilation (2, 2) from Dumoulin et al., 2018. The dilation spreads the receptive field of the kernel by inserting gaps in between its elements.

particularly significant.

Substantial effort has been made to explain how deep neural networks make their predictions. Many of the popular methods, such as maximal activating inputs (Erhan et al., 2009), class-activation maps (Zhou et al., 2015), saliency maps (Simonyan et al., 2014) or layer-wise relevance propagation (Sturm et al., 2016) have been used also in interpreting DNNs used for brain signal decoding (Goodfellow et al., 2018; Hartmann et al., 2018; Rieke et al., 2018; Sturm et al., 2016; Yang et al., 2018).

2.3 DNNs for (i)EEG decoding

In this section, we introduce the Deep4Net proposed by Schirrmeister et al., 2017 which is the main focus of the presented experiments. Then we describe another paper - by Hammer et al., 2021 - who used the Deep4Net to solve the same task as we are solving in this thesis on the same data and laid the foundation of the here presented research.

2.3.1 Schirrmeister et al.

The paper by Schirrmeister et al., 2017 introduced multiple CNN and hybrid architectures suitable for movement decoding from raw EEG signals, including the Deep4Net. It has several significant contributions. First of all, it shows that CNNs are able to achieve at least as good performance on classification from raw EEG signals as the widely used Filter bank common spatial patterns (FBCSP) do from manually extracted features. Besides that, it shows that recent advances in machine learning such as batch normalisation or the exponential linear unit (ELU) activation function help boost the decoding accuracies of networks used for movement decoding tasks. Lastly, using a perturbation visualisation method they explore, which spectral power features are important for the network's predictions. They show a substantial effect of frequencies in the alpha and beta band. They also observe an effect of the high-gamma frequencies on the predictions, which, they claim, is for the first time when using non-invasive EEG data.

2.3.2 Hammer et al.

In 2021 Hammer et al., 2021 used the Deep4Net architecture for regression on intracranial EEG. Differing from the original Deep4Net paper in the type of task and the data type, they inspected the performance and features of the architecture in these altered settings. They visualised the important spectral features investigating learnt specialisation of single network units to either phase or amplitude. This thesis directly follows up on their research.

Regression task and performance

In this paper, the Deep4Net was used to predict kinematic variables, namely velocity and absolute velocity (speed), based on intracranial EEG of subjects undergoing pre-surgical epilepsy screening. The subjects were asked to play a simple video game. Their task was to use a joystick to control a car moving forward on a winding track. The joystick allowed for left and right movement, and the velocity and absolute velocity of this movement were recorded along with concomitant iEEG signals. The dataset and the task are described in more detail in Section 3.1.4.

Unsurprisingly, the Deep4Net significantly outperformed linear regression used as a baseline on this task, especially for absolute velocity, as is apparent from Figure 2.4.

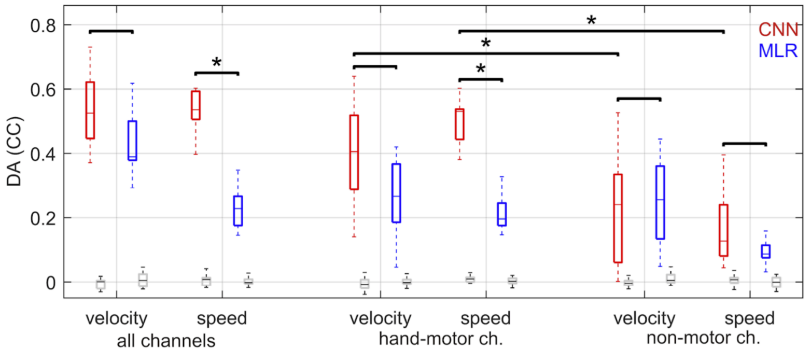


Figure 2.4 "The decoding models for both the Deep4Net (CNN) and multi-linear regression (MLR) used iEEG data from all channels, hand-motor channels, or non-motor channels (see Section 3.1.3). The performance was assessed by correlation coefficients (CCs) between predicted and real kinematic parameters. In each box-plot, the box indicates the interquartile range (IQR) of the DA distribution across participants, the median is marked by a horizontal line inside the box, the whiskers indicate 1.5-times the IQR. The chance-level CCs for shuffled datasets was assessed for both CNN and MLR (grey box-plots below)" Hammer et al., 2021

Training regime

The Deep4Net was trained using the ADAM optimiser with a learning rate 0.01 for 100 epochs. The mean squared error (MSE) was used as the loss function to calculate the error of the network's predictions. A leave-one-out-cross-validation on 25 second long data segments (see Section 3.1.4) was employed to estimate performance. The goodness of prediction was evaluated on the one fold not included in training using the linear Pearson's correlation coefficient (Pearson et al., 1895). Chance level performance was assessed by randomly pairing the inputs and gold labels. For example, iEEG data from fold 2 were paired with gold kinematic data from fold 6.

Input perturbation - output correlation visualization

The perturbation analysis in Hammer et al., 2021 was more thorough than in the case of Schirrmeister et al., 2017 Besides looking at the influence of amplitude perturbations, they also investigated the influence of phase perturbations, similarly to Hartmann et al., 2018. Moreover, attention was paid to the specialisation of single units to either phase or amplitude. While most findings were in line with previous literature, the low sensitivity of the network to high-gamma frequency bands amplitudes was unexpected. Especially considering two things: 1. This same architecture was able to use high-gamma information in EEG decoding where high frequencies contain more noise than in iEEG signal in Schirrmeister et al., 2017. 2. High-gamma has been shown to be informative for absolute velocity decoding on a subset of the same data when using linear regression in Hammer et al., 2016.

2.3.3 Related work

Besides the Deep4Net, other networks were designed to decode EEG signals. One of these networks is the EEGNet from Lawhern et al., 2018. The authors build a CNN architecture, which is able to generalise over different BCI paradigms. They show that the EEGNet can learn a wide variety of interpretable features over a range of different EEG decoding tasks, including movement related cortical potentials and sensory-motor rhythms.

An architecture idea recently repeatedly used for decoding from EEG and iEEG signals is combining recurrent layers and convolutional layers to form hybrid networks. Examples of utilising such networks in movement-related paradigms are, for example, Xie et al., 2017; Zhang et al., 2019. In Xie et al., 2017 a network constituting of 4 convolutional layers followed by one long-short term memory (LSTM) (Hochreiter et al., 1997) layer was utilized to decode individual

finger flexion. In Zhang et al., 2019, three convolutional layers followed by three LSTM layers were utilised to solve the BCI IV 2a competition task (same as in Schirrmeister et al., 2017).

Chapter 3

Methodology

In this chapter, we go over the details and specifics of the dataset and the methods used in this thesis. We first provide information related to the dataset (Section 3.1) and then give a detailed description of the Deep4Net architecture (Section 3.2.1) together with information about the training regime (Section 3.2.4). We also describe the gradient visualization method which we chose to interpret the decisions of the CNNs (Section 3.2.5).

3.1 Dataset

The dataset was the same as in the unpublished study of Hammer et al., 2021. Modified with the authors permission, the description of the experiment settings and the pre-processing follows.

3.1.1 Movement task and kinematic variables

A dataset that was already used in several other iEEG studies to examine decoding of movement kinematic parameters (Hammer et al., 2013, 2016, 2021) was utilized for the purposes of this thesis. The participants performed a motor task-based on driving a car in a computer simulation. They controlled the car's position on a computer screen using a steering wheel that they held in both hands. The task was to keep the car on a curved road. The road was random without any repetitions, following a low-pass filtered white noise trajectory. During the movement task, the position of the car on the road was measured. The position of the car linearly corresponded to the deflection of the steering wheel and was measured relative to the screen centre, which corresponded to zero-deflection of the steering wheel. Notably, the control of the car's position was possible only in the horizontal dimension (left-right). The upward movements of the car (ver-

tical scrolling speed) were kept constant in each run of the game and adjusted for each subject individually.

The difficulty and the recording time of each subject were also adjusted based on the participant's motivation/ability to participate, lasting 25 $\pm$ 7 min (mean $\pm$ SD). The car game difficulty was modified by the vertical scrolling speed of the car. Therefore, to account for faster movements, the low-pass cutoff frequency was set to 10 Hz for smoothing the raw tracker data before the derivation of the kinematic parameters.

From the horizontal, 1-D trajectory, the following two kinematic parameters were derived: 1. velocity computed as a derivative of position, 2. speed as the absolute value of velocity. Velocity thus contained the directional information in its sign; specifically, velocity values smaller than zero implicated movement to the left and vice versa, while speed indicated how fast the car was moving left or right (irrespective of its direction). The time-series of the kinematic parameters were resampled at 250 Hz and temporally aligned to the iEEG data.

3.1.2 Recording

The recordings were performed in the University Medical Centre in Freiburg, Germany and in the Motol University Hospital in Prague, Czech Republic. The study included 12 epilepsy patients (6 male, age 19–50, 33 $\pm$ 10, (mean $\pm$ SD), all of which had intracranial EEG implantations. Some of the implantations were placed in the region of the motor cortex. The location of the electrodes was dependent solely on the needs for medical evaluation of their medication-resistant epilepsy. Both sEEG and ECoG electrodes were present among patients. Detailed information about electrode type and placement is presented in Table 3.1 and Figure 3.1.

3.1.3 Separation of motor and non-motor channels

The separation of the channels was not a part of this thesis. We already obtained separated channels. The specific separation criteria are described below as provided by Hammer et al., 2021.

One of the aims of the Hammer et al., 2021 study was to show what the CNNs learned from the raw brain signals. The hypothesis was that the CNNs would focus on information from the hand-motor cortex when solving a movement-related task. The recorded channels were divided into two distinct, non-overlapping groups: 1. hand-motor channels and 2. non-motor channels to verify this hypothesis. The hand-motor channels induced a clear hand motor response after the electrical stimulation at low intensities underneath or around

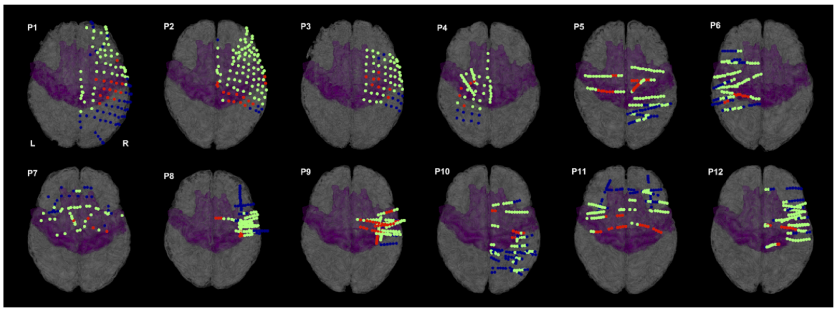


Figure 3.1 Implantation schemes of the 12 patients (P1 - P12) taken from Hammer et al., 2021. The motor cortex is highlighted in magenta (defined as the union of areas 4 and 6 of the SPM Anatomy Toolbox). The electrodes are color-coded in the following way: Red - channels with hand-motor response after electrical stimulation mapping; dark blue - non-motor channels; green - other channels.

Patient	Sex (Age)	Pathology	Implant loc	Electrode type (N)	N_r	N_{ch}	N_m	N_{nm}
P1	F (47)	CG	Right frontal	ECoG (98), sEEG (12)	25	85	14	47
P2	M (48)	FCD	Right fronto-temporal	ECoG (90), sEEG (12)	53	49	15	11
P3	M (50)	CG	Right frontal	ECoG (64)	3	61	8	11
P4	F (22)	FCD	Left frontal	ECoG (40), sEEG (12)	11	47	5	16
P5	F (29)	Gilosis	Left and right fronto-parietal	sEEG (125)	10	115	23	29
P6	M (29)	FCD	Left frontal	sEEG (125)	34	91	9	39
P7	F (32)	FCD	Left and right insular	sEEG (61)	14	47	4	22
P8	M (19)	FCD	Right insular	sEEG (117)	16	101	9	36
P9	F (25)	Gliosis	Right insular	sEEG (117)	21	96	35	12
P10	M (34)	FCD	Right fronto-parietal	sEEG (125)	23	102	8	63
P11	M (34)	FCD	Left and right frontal	sEEG (126)	11	115	21	48
P12	M (28)	Not operated	Right frontal	sEEG (125)	20	105	10	27

Table 3.1 N_r, N_{ch}, N_m, N_{nm} : number of rejected, non-reject, hand-motor, and non-motor channels, respectively. FCD: focal cortical dysplasia; CG: Cryptogenic.

the electrode contacts. The non-motor channels, on the other hand, produced no sensory/motor response after the stimulation. Furthermore, a requirement of at least 1 cm distance from the motor and pre-motor areas (i.e. Area 4a, Area 4p and Area 6 from the SPM Anatomy toolboxEickhoff et al., 2005) had to be satisfied for a channel to be included in the non-motor group. To this end, all MRI brain scans (T1-weighted sequence) were normalized to the MNI space and the electrodes' coordinates were read out from either the post-implantation MRI or CT scans53.

The average number of hand-motor channels was 13 $\pm$ 9 (mean $\pm$ SD), while there were 29 $\pm$ 18 (mean $\pm$ SD) non-motor channels. Some electrodes did not fall into either the hand-motor or non-motor channel groups (e.g. channels in the motor cortex, the electrical stimulation of which induced leg-motor response). These electrodes were then left out in some analysis because the aim was to delineate the difference between the two distinct groups of channels (the hand-motor channel group and the non-motor channel group clearly far away from the motor cortex).

3.1.4 IEEG data preprocessing

The iEEG data pre-processing was also already completed when we obtained the dataset, and it was not carried out as a part of this thesis. Nevertheless, we provide the description of pre-processing from Hammer et al., 2021: A comprehensive rejection of *epileptic* channels based on the information from the respective epilepsy centres was performed because the primary aim was to investigate the physiological brain activity. Thus, the epileptic channels, i.e. those located in the seizure onset zone and/or containing a large number of interictal epileptiform discharges, were rejected from this study (20 $\pm$ 13, mean $\pm$ SD) over subjects; see Table 3.1. All non-rejected iEEG channels (85 $\pm$ 28, mean $\pm$ SD) were referenced to their common average (CAR), high-pass filtered at 0.15 Hz (3rd order Butterworth filter), normalized to the inter-quartile range of each channel, and resampled to 250 Hz, to yield consistent data sets from the different recording systems used at both aforementioned epilepsy centers[1]. The iEEG data were resampled to 250 Hz to emulate the same setup of the Deep4Net from Schirrmeister et al., 2017, which was successfully applied to demonstrate high-gamma (70 - 90 Hz) effects in decoding motor behaviour from non-invasive EEG. Importantly, any over-fitting in the pre-processing was carefully avoided (i.e., all parameters of the pre-processing of the test set were estimated on the training set) and only causal, finite impulse response filters were applied. Therefore, the decoding ap-

[1]Motol University Hospital and University Medical Centre Freiburg

proach could be readily applied also in a closed-loop, online BMI.

The aligned iEEG and kinematic time-series were divided into 25-s long data segments. In order to minimize the potential influence of temporal correlations in neighbouring data parts, the segments had a 2-s margin in between each other. Additionally, the last two minutes of the recordings were left as a test set. The test set was not a part of the dataset that was utilized in this thesis.

3.1.5 Multiple dataset conditions

In this thesis, we refer to multiple variations of the original dataset provided by the Motol University Hospital. Following modifications were explored: 1. filtering out certain frequencies, 2. shifting the predicted time-point with respect to the input window, 3. spectral whitening.

- **Filtering** We created two types of datasets using filtering. A *high-pass filtered dataset* using a 15th order Butterworth filter, cut-off frequency 60 Hz, non-zero phase shift and a *low-pass filtered dataset* using a Butterworth filter order 15, cut-off frequency 40 Hz, non-zero phase shift. Besides full training on these datasets, parts of these datasets were combined for training and validation to see how a network trained on full data performs on high-passed data etc.

- **Shifting** Originally in Hammer et al., 2021 the dataset was constructed so that the predictions were made from signals recorded prior to their execution (causal prediction). In addition to this, we also created datasets where the labels (i.e. values of kinematic variables) were shifted so that predictions were also made from signals recorded after movement execution (acausal prediction). While a network trained in this manner is unsuitable for online BCI, it allows us to inspect which time frame of the signals contains information about the movement. More details about how the shift influences the predictions and why can be found in Section 3.2.3.

- **Spectral whitening** Lastly, we also created whitened datasets. Spectral whitening normalizes amplitudes of all frequencies to one. Using a Fourier transformation of each 25s long segment (which the dataset was divided to) into the power spectrum, obtaining information about phase and amplitude was achieved. Then dividing the amplitudes with their absolute value and then, via inverse Fourier transformation, transforming it back to signal. The whitened signal was normalized to its interquartile range (IQR). This normalization was calculated and applied to each segment on

the training set, and the same parameters averaged over all 25s long segments of the training set were applied to the validation set. Figure 3.2 illustrates the effects of spectral whitening on a portion of one of the 25s long segments.

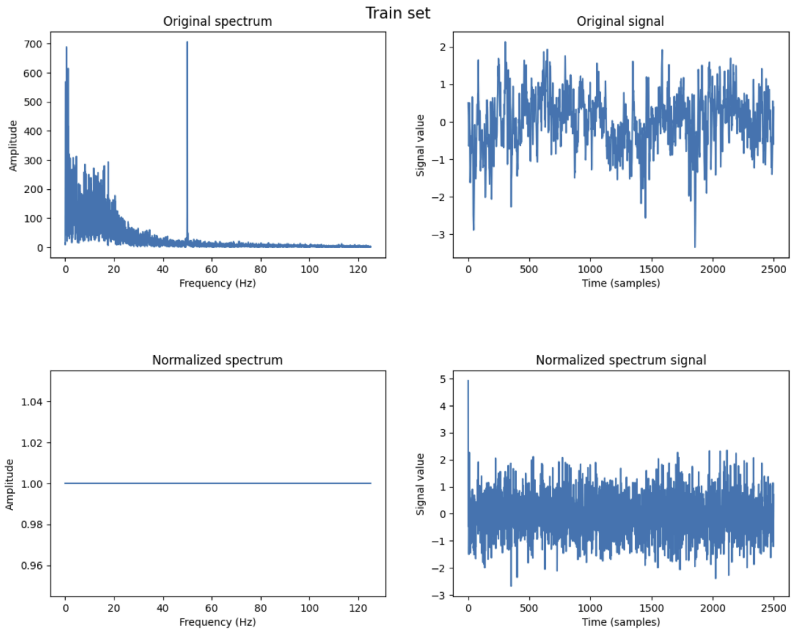


Figure 3.2 The top row shows the original power spectrum of the iEEG signals and the original iEEG signals of 10 seconds of the first 25s long segment. The bottom row shows how spectral whitening changed both the spectrum and the signals.

3.2 Deep4Net

The Deep4Net, which is freely available in the Braindecode library[2], was already shortly described in Section 2.3.1. In this section, we describe it in more detail. Besides describing the architecture as used in Hammer et al., 2021 we focus on how cropped decoding is used (Section 3.2.2), what the receptive field of the network looks like and how it affects the predictions (Section 3.2.3), we describe the training procedure (Section 3.2.4), the visualization of the network's gradi-

[2]www.braindecode.org

ents (Section 3.2.5) and the architectural modifications we introduce in this thesis (Section 3.2.6). Lastly, we also state how performance is analyzed Section (3.2.7).

3.2.1 Architecture

The architecture has been previously used in a number of EEG decoding tasks (Hammer et al., 2021; Hartmann et al., 2018; Schirrmeister et al., 2017). It is implemented in the PyTorch framework[3]. The input of the network is a 2D array with time-steps along one axis and channels along the other axis. The architecture consists of four convolutional-max-pooling blocks and one final convolutional layer. It is designed so that a special first block can learn spatially global filters. The following three standard blocks (conv_2 - conv_4) then allow for learning temporal hierarchies of local and global modulations (Schirrmeister et al., 2017). The first convolutional block is split into two parts. One performs convolution over time (conv_temp) and the second over the channels with weights for all possible electrode pairs using filters of the preceding temporal convolution (conv_spat). Because there is no activation function between these two layers, they could be merged, but the authors emphasize the regularization function of this separation because it forces a separation of the linear transformation into a temporal convolution and a spatial filter.

A convolutional block of the Deep4Net starts with a convolutional layer, then a batch normalization layer follows, after which a non-linearity is added; in the case of the Deep4Net, it is the exponential linear unit (ELU) function. Finally, a max-pool layer closes the convolutional block. An output layer, which initially was a softmax, comes last. , Only a single modification needs to be done to use this network for the regression task the present thesis focuses on: removing the last softmax layer and replacing it with a convolutional layer. This last layer is called the conv_classifier. The architecture already transformed for regression is depicted in Figure 3.3.

The Deep4Net is able to process varying input lengths. In each convolutional or max-pool layer, it simply slides its filters over the input and gives a respective number of outputs. Therefore, before training, the number of outputs needs to be calculated based on the input window size that was chosen. The corresponding values (gold labels) of the predicted kinematic variables are cropped accordingly. In this thesis, we used 1200 samples as input window length (which corresponds to 4.8 seconds) to be consistent with Hammer et al., 2021. The Deep4Net as used in Hammer et al., 2021 gives 679 predictions (Figure 3.3).

Lastly, we point out that the network does not use any padding. No padding makes its receptive field non-uniform; we discuss the consequences in Sec-

[3]https://pytorch.org

tion 3.2.3. Nevertheless, it is necessary to avoid padding to perform cropped decoding described in Section 3.2.2.

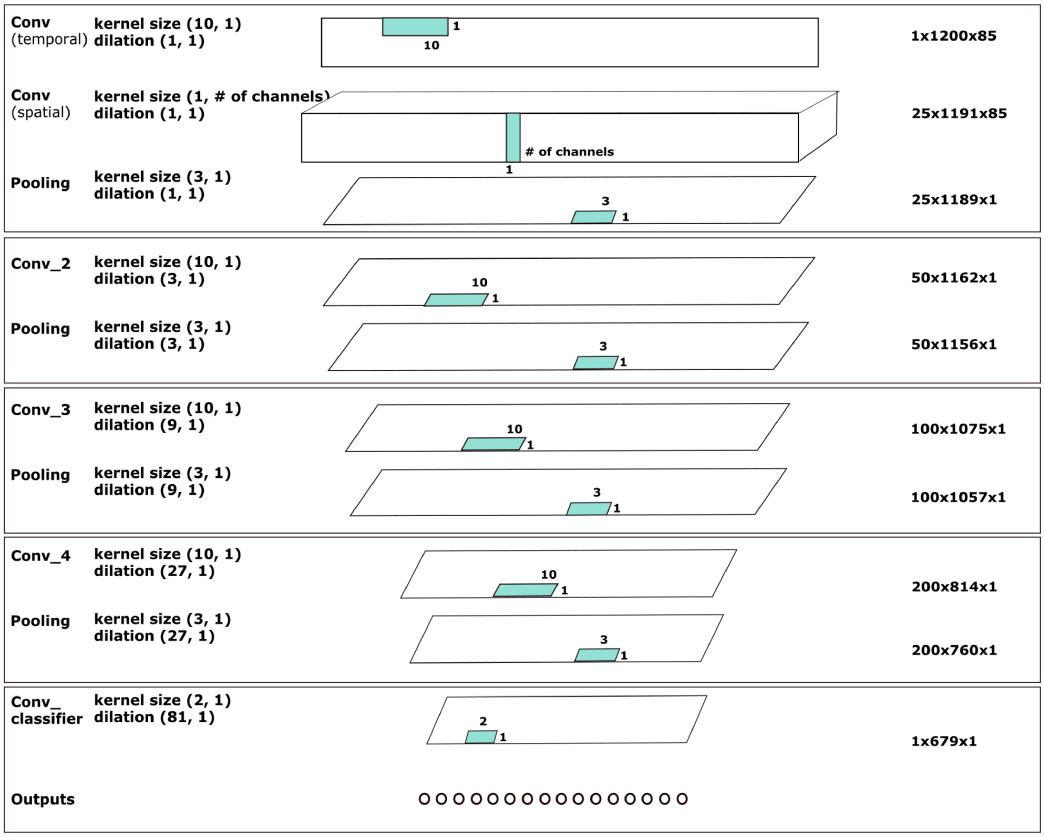


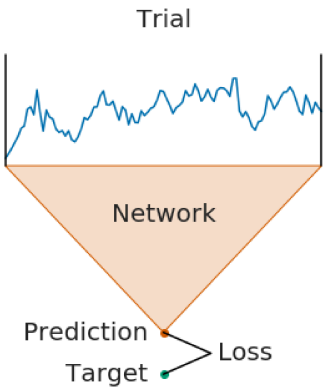
Figure 3.3 The architecture of the Deep4Net modified to be suitable for regression as used in Hammer et al., 2021 and in this thesis as k3_d3_sbp0.

3.2.2 Cropped decoding

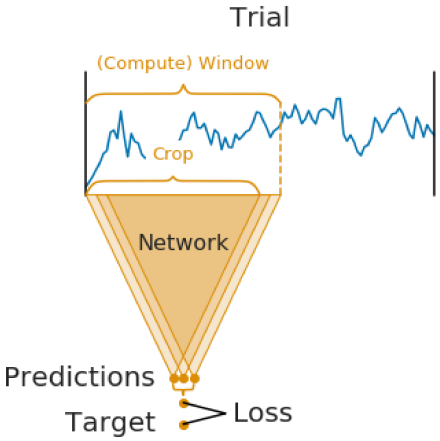
The possibility to use cropped decoding is implemented in the Braindecode library together with the original Deep4Net. Cropped decoding allows, instead of obtaining one prediction per trial (Figure 3.4a), to separate the trial window into multiple smaller overlapping sub-windows (crops), each of which produces a prediction (Figure 3.4b).

It is possible to implement cropped decoding as processing one crop at a time. It can be sped up with dense prediction (discussed in Section 2.2.1). Every CNN[4], which processes one crop at a time (one-crop network) and has no right padding,

[4]By every we mean standard CNN architecture similar to the Deep4Net consisting of convolution, pooling and batch-normalization layers.



(a) Trial-wise decoding



(b) Cropped decoding

Figure 3.4 This Figure obtained from the Braindecode library documentation[a] illustrates the difference between cropped decoding and trial-wise decoding for classification.

[a] https://robintibor.github.io/braindecode/notebooks/Cropped_Decoding.html

can be transformed into a dilated network that simultaneously processes multiple crops and gives outputs equivalent to the one-crop network. A visual explanation of this is in Figure 3.5.

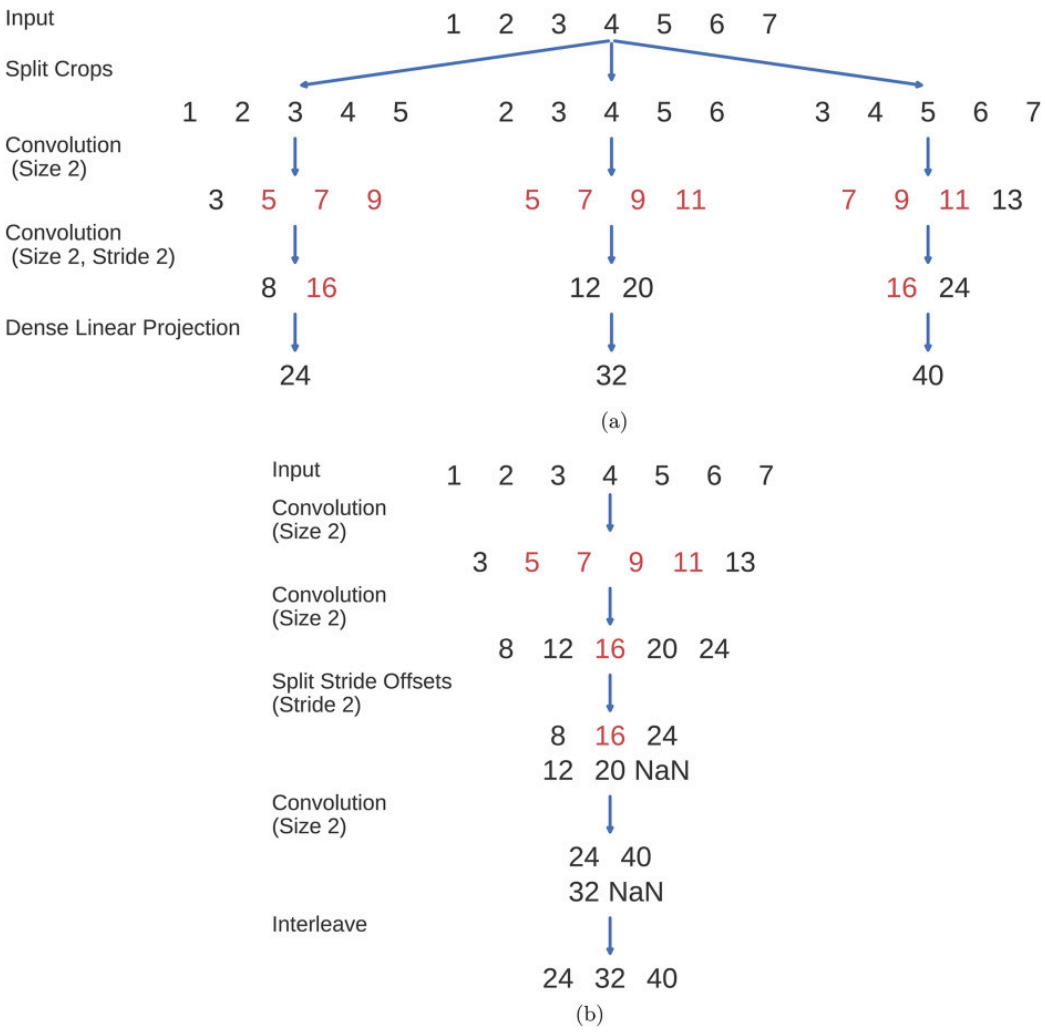


Figure 3.5 A toy example from Schirrmeister et al., 2017 presenting the difference between **a)** the naive approach to process crops using a network with strides; and **b)** processing multiple crops at the same time using the dilated network.

The transformation of the one-crop network into the corresponding dilated network sets the strides in convolutional and max-pooling layers to one and replaces them with increased dilations. The algorithm used to make the transition between strides and dilations is described in Algorithm 1. This algorithm allows us to transform every one-crop network without right padding into an equivalent dilated network.

Nevertheless, this transformation relationship is not symmetric. For some dilated networks, no one-crop network with strides (without right padding) exists. An example of such a network is a CNN wherein two consecutive layers with a dilation parameter the first layer has bigger dilation than the second one. A bigger dilation followed by a smaller one cannot occur in a network build by following Algorithm 1 thus, such a CNN was not derived from a one-crop network.

Algorithm 1 The simplified algorithm used to transform a network with strides to a network with dilations in the Braindecode library. It assumes a 1D stride and dilation. 1D stride and dilation are sufficient in our case because all the strides in the networks are 1D.

```
def to_dense_prediction(network):
  current_stride = 1
  for module in network.modules:
    if hasattr(module, 'dilation'):
      module.dilation = current_stride
    if hasattr(module, 'stride'):
      current_stride *= module.stride
      module.stride = 1
```

In Schirrmeister et al., 2017 where the Deep4Net originated, the architectures were constructed as one-crop networks. They were then converted into dilated networks. This leaves a whole set of architectures unexplored: those dilated networks which cannot be constructed from a one-crop network.

In this thesis, we use the dilated networks to obtain multiple values of kinematic variables from multiple crops processed at the same time. Because there is no reason for our task to consider one-crop networks first and then create dilated networks from them, we are free to explore these architectures. And we do so in Section 4.1.

3.2.3 Receptive field

The fact that the Deep4Net cannot have any right padding to process multiple crops simultaneously affects its receptive field. By receptive field, we mean the part of the network's input, which affects one specific output (in our case, it corresponds to one crop from which one prediction is made). In Figure 3.6 we visualize the receptive field used to predict one time-point. The figure shows how many times each input sample, and the values derived from it, are used in a forward pass through the CNN to obtain the specific prediction. Note that the receptive field in no way illustrates the weights of the network.

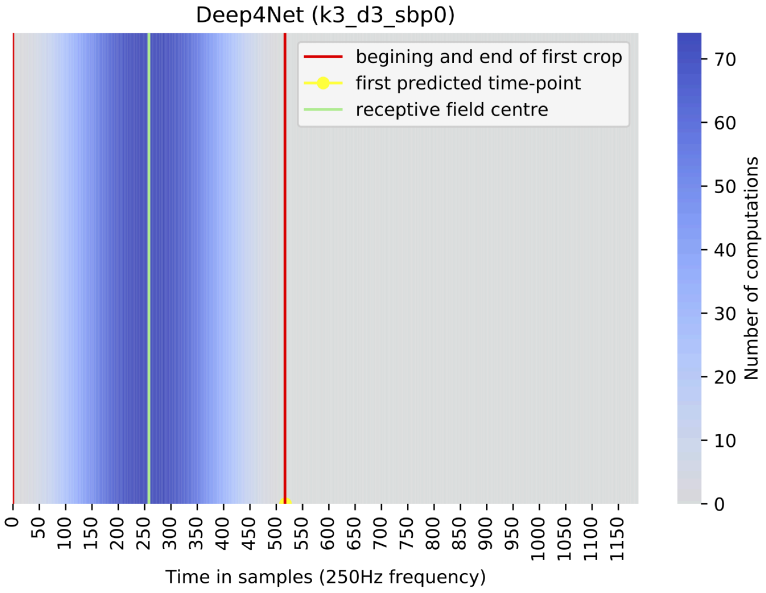


Figure 3.6 The receptive field of the original Deep4Net, which is the base model derived from Hammer et al., 2021, in this thesis denoted as variable_k3_d3_sbp0. The vertical red lines delimit the receptive field (crop) of the CNN, which is used for calculating the first prediction. The whole grey area represents one input window. The green line represents the centre of the receptive field, the yellow dot on the x-axis the first time-point that is being predicted. The gradient bar on the right shows the number of times each input sample was used in the output calculations.

Figure 3.6 illustrates that the predicted time-point is fairly far from the centre of the receptive field. This is because we make the prediction only based on the data recorder prior to the predicted movement and do not use padding. The CNN mainly focuses on signals around the centre of the receptive field and, therefore, is biased towards data that are far from those that directly precede the movement. Another illustration of the relation between the receptive field and the predicted time-point is in Figure 4.6, **Graph A**.

For classification, this is not as critical because the same label is predicted from multiple crops, and the actual classified movement does not happen at the end of the last crop but at some point during the whole input window from which the crops are derived (Schirrmeister et al., 2017). This gives the network a chance to focus on information directly preceding the actual movement.

We focus on the changing role of the receptive field between classification and regression because, as previously stated in Schirrmeister et al., 2017 and Hartmann et al., 2018, this same CNN architecture was shown to utilize information from the high-gamma frequency bands. Since we are investigating the frequency bands and their effective utilization, we inspect this difference as a possible reason why the network, when employed to predict velocity and absolute velocity of hand movement, does presumably not use information from the high-gamma frequency band. The hypothesis here is that the high-gamma signal is most informative directly before the predicted movement execution. Therefore, the distance of the network's receptive field centre to the predicted time-point hinders its usage. We test this hypothesis in Section 4.2.

3.2.4 Training

We left all hyper-parameters equal to those from Hammer et al., 2021 because it was our goal to reproduce their results and build upon their research. The only parameter that we changed was the learning rate. The network was trained for 100 epochs with batch size 32, using the ADAM optimizer and learning rate 0.001. The reported learning rate in Hammer et al., 2021 was 0.01. We used a lower learning rate because otherwise, the results were significantly worse than reported.

In Hammer et al., 2021 they performed leave-one-out cross-validation on the 25s-long segments. Nevertheless, with the amount of different architecture and dataset modifications we explored, this would be extremely time-consuming. 5-fold-cross-validation gave us a good estimation of the networks' performances while also saving computational power. To perform a 5-fold-cross-validation, we divided the 25s-long segments into five equal or almost equal subsets. For each patient, a separate 5-fold cross-validation was conducted. The final performance was estimated from all five folds over each patient (see Section 3.2.7).

3.2.5 Gradient visualization

The gradient visualization algorithm is the following. First, the input signal is converted into the power spectrum using Fourier transformation. After this transformation, hooks are created for the amplitude and phase values. Then it is converted back into the original signal, this time using the `torch` functions so PyTorch can start building its computational graph reaching back to the amplitudes and phases. This tracked signal is then given to the network, whose weights are frozen, as input. After obtaining the outputs, they are averaged and the `torch.autograd.backward()` function is used to derive the gradients with respect to the frequency amplitudes and phases (Simonyan et al., 2014).

The above-described procedure was always repeated over multiple batches for each patient, and the resulting gradients were averages over the batches to obtain representative gradients. Gradients are useful in showing to which frequencies' modulations the network reacts strongly. The higher the gradient value for a certain frequency, the more the change is this frequency's amplitude influences the prediction.

As described in Section 3.2.1 the network can process varying input window lengths. With a change in input window length, a change in the number of outputs occurs. Throughout the thesis, the input window length used in the above-described algorithm was set to twice the size of the receptive field of the network to be consistent with Hammer et al., 2021. A special case is the gradient visualization when the input window is shortened so that the network has only one output was also briefly explored as a part of this thesis in Section 4.1. In this special case, the outputs do not have to be averaged before using the `torch.autograd.backward()` function to derive the gradients.

We chose to always display the gradient values for the training set because there are no significant differences between the gradients of the networks on the training set and the validation set. Also, note that in this thesis, we only address the gradients for the amplitudes of frequencies. While inspecting the phase gradients would also be interesting, it is out of the scope of this thesis.

3.2.6 Architectural modifications

During our research, we made a set of modifications to the architecture of the Deep4Net, creating multiple new CNNs. We focused on the kernel sizes and dilation parameters of the four max-pool layers present in the network with our changes. Besides the original kernel sizes of the max-pool layers, which were set to (3, 1) for all layers, we also explored kernel sizes (2, 1) and (1, 1). The dilations in the original network were (1, 1), (3, 1), (9, 1), (27, 1) for the four max-pool layers. We considered also dilations: 1. powers of three: (3, 1), (9, 1), (27,

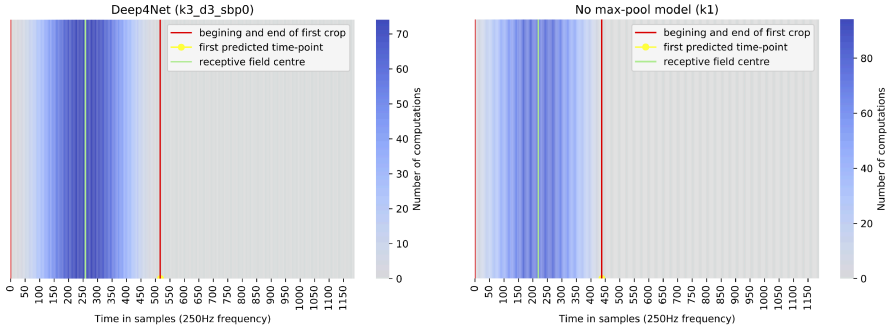
1), (81, 1); 2. powers of two: (2, 1), (4, 1), (8, 1), (16, 1); 3. powers of one: (1, 1), (1, 1), (1, 1), (1, 1). Because the second dimension of the kernel sizes and max-pool layers is always one, from now on, we only specify the first dimension, and when not said otherwise, the second dimension is one. Table 3.2 explains the terminology used concerning the changing max-pool parameters used throughout the thesis. The original Deep4Net corresponds to the {variable}_k3_d3_sbp0 in Table3.2. The suffix `sbp` refers to a parameter of the Deep4Net, namely the `stride_before_pool` parameter. This parameter influences the dilations in the max-pool layers of the network. In Hammer et al., 2021 and Schirrmeister et al., 2017 this parameter was set to `False (sbp0)`. The only analysis where the `stride_before_pool` was set to `True (sbp1)` was when analyzing a gradient peak (briefly discussed in Section 4.1) and also to study the dependence of the size of the receptive field on the performance of the network (Figure 4.7). Therefore, we mention the Deep4Net with `sbp1` in Section 4.1 but the default Deep4Net architecture is the one with `sbp0` as in Hammer et al., 2021 and Schirrmeister et al., 2017. We do not specify the `stride_before_pool` parameter for the other CNNs we created. It becomes meaningless because we manually set the sizes of the dilations for all the max-pool layers with no consideration of this parameter.

When we change the parameters of the max-pool layers as we describe above, the number of outputs of the network changes. It also causes changes in the receptive field of the network. To see how, consider just one max-pool layer with dilation 3 (L3) and one max-pool layer with dilation 1 (L1), both without padding. If we use windows of the same length as input for L3 and L1 and compare their outputs, the output dimension of L3 will be smaller than of L1. Consequently, if we alter the dilation (and kernel size) parameters of the max-pool layers in the Deep4Net, we get a different number of outputs from the altered network.

When one network has dilations 1, 3, 9, 27 in the max-pool layers (the k*_d3) and the other 1, 1, 1, 1 (k*_d1), the difference in the dimension of the output is significant. The latter network makes more predictions from the same input window length because the output dimension is larger. It also has a smaller receptive field for one prediction, as is visualized in Figure 3.7. Information about the size of the receptive field can be found in Table 3.2.

3.2.7 Performance analysis

When comparing two CNNs, we are not comparing them based on the mean-square error (MSE), which is the loss function of the training, but on the correlation coefficient (CC) between the model's predictions and the predicted kinematic variable on the validation set. Specifically the Pearson's correlation coefficient (Pearson et al., 1895). Often throughout the thesis, we show boxplots of CCs of different architectures. Each boxplot is obtained from performing a



(a) Receptive field of the Deep4Net (k3_d3_sp0).

(b) Receptive field of the CNN without max-pool (k1).

Figure 3.7 Both graphs **(a)** and **(b)** show the receptive fields (delimited by red lines) used for predictig the first time-point (yellow dot on the x-axis). The receptive field in **(b)** belongs to a network with dilations 1 in all max-pool layers. It is smaller, reaching less than 450. The receptive field of the original Deep4Net in graph **(a)** is larger; it reaches beyond 500. This difference is caused by the different kernel sizes and dilation parameters of the max-pool layers.

Model name	Max-pool kernel size	Max-pool dilations	Stride before pool	Size of the receptive field
variable_k1	1, 1, 1, 1	–	–	442 samples
variable_k2_d3	2, 2, 2, 2	3, 9, 27, 81	–	562 samples
variable_k3_d3_sbp1	3, 3, 3, 3	3, 9, 27, 81	True	682 samples
variable_k3_d3_spb0	3, 3, 3, 3	1, 3, 9, 27	False	522 samples
variable_k2_d1	2, 2, 2, 2	1, 1, 1, 1	–	446 samples
variable_k3_d1	3, 3, 3, 3	1, 1, 1, 1	–	450 samples
variable_k2_d2	2, 2, 2, 2	2, 4, 8, 16	–	472 samples
variable_k3_d2	3, 3, 3, 3	2, 4, 8, 16	–	502 samples

Table 3.2 This table gives an overview of the various CNN architectures we investigated in the scope of this thesis. Modifications were always made to the max-pooling layers. Only information about the first dimension of the kernel sizes and dilations is presented. The second dimension is always 1. I.e. max-pool kernel sizes 2, 2, 2, 2 actually represent kernel sizes (2, 1), (2, 1), (2, 1), (2, 1).

5-fold-cross-validation for each of the 12 participants. The validation set CCs from the five folds are averaged, and the boxplot is created from the 12 averages. While it is possible to create the boxplots from all the folds, not averaging them for each patient, we are interested in the distribution of correlation among patients rather than among individual runs.

Chapter 4

Experiments and results

To reach our goals, we will utilise three types of experiments on the Deep4Net model: 1. we will systematically change the parameters of the Deep4Net max-pool layers and assess change in performance and gradients (Section 4.1); 2. we will shift the predicted time-point across the receptive field of the network and assess the performance and gradients of the architectures defined in the previous step (Section 4.2); 3. we will perform spectral whitening on the dataset and assess how it influences the performance and gradients of the established architectures (Section 4.3).

The code for the experiments is included in the thesis attachment, and it is also available at `https://github.com/Mvystrcilova/ECoG_decoders`.

4.1 Architectural modifications

In Hammer et al., 2021 their input perturbation visualisation technique did not show the expected effect of modulations in the high-gamma frequency bands on the network's predictions. Without including it in their paper, they studied it further using a different visualisation technique, namely the gradient visualisation (see Section 3.2.5). They found that this visualisation technique also does not show any interest of the Deep4Net in the high-gamma frequency band unless the input window is shortened so that the CNN has only one output, i. e. predicts one time-point[1]. In such a scenario, a significant gradient peak occurred at 83.33 Hz.

We adopted their visualisation technique and investigated this gradient peak

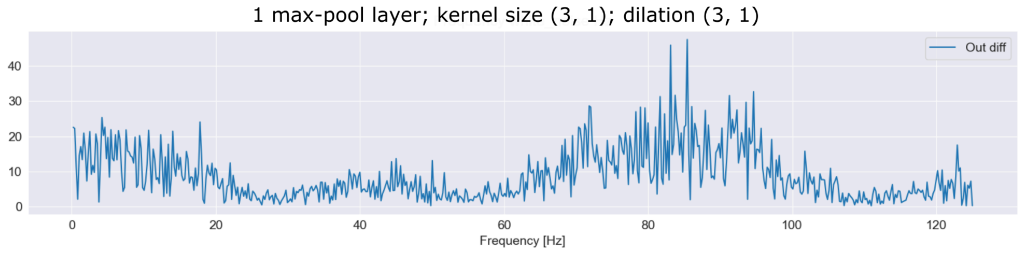
[1]Because the network has no layers accepting only a fixed number of inputs, it is able to process varying input lengths. It simply slides its convolutional and max-pool kernels over the input giving the corresponding number of outputs. Therefore, it is possible to set the input size so that the network only has one output. See Section 3.2.5 for more details.

further. We have hypothesised that the 83.33 Hz peak is an artefact of the network's architecture. We have thus performed a series of experiments, where we have modified the Deep4Net architecture parameters, and studied the influence of these architecture parameters changes on the network's gradient profile. Specifically, we changed the kernel sizes and dilations of the max-pool layers. We explored kernel sizes 1 (i. e. kernel size 1, 1, 1, 1 for the four max-pool layers present in the network), 2 (i.e. kernel sizes 2, 2, 2, 2) and 3 (i.e. kernel sizes 3, 3, 3, 3) and dilations powers of 1 (i.e. dilations 1, 1, 1, 1 for the four max-pool layers present in the network) powers of 2 (i.e. dilations 2, 4, 8, 16) and powers of 3 (i.e. 3, 9, 27, 81). Detailed information about the different architectures can be found in Table 3.2.

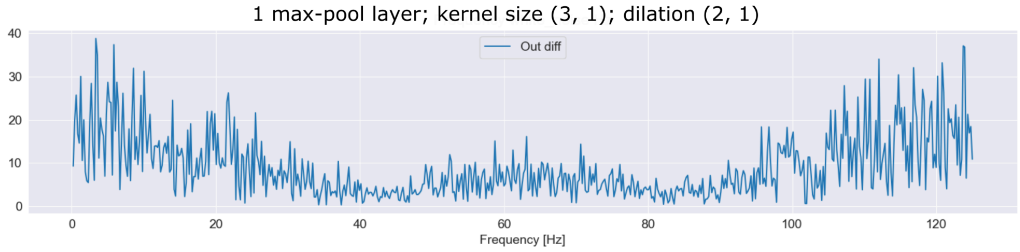
These experiments concluded that the 83.33 Hz peak is indeed an architectural artefact. When changing max-pool layer dilations in the whole network from powers of 3 to powers of 2 or 1, the gradient peak disappeared. It disappeared independently on the kernel sizes and without a decrease in performance. A detailed analysis of performances of the architectures is presented in Section 4.1.1.

We think that the peak occurred due to frequency alignment between the sampling rate (250 Hz) and the dilation of the max-pool layers (which are powers of three) because $250/3 = 83.33$. This frequency alignment can be visualised when passing the signal through a single max-pool layer. Figure 4.1 shows the difference between the output of one max-pool layer and the output of one max-pool layer when white noise is added to the input. In Figure 4.1a the max-pool layer has dilation 3 and kernel size 3. In Figure 4.1b the max-pool layer has dilation 2 and kernel size 3. It is clear that even though white noise contains all frequencies, an increase in only some frequencies was projected on the output. We can notice that there is a periodicity among the frequencies which increase most in the output. This period depends on the dilation. The most prominent peak for the max-pool layer with dilation 3 is around 83.33Hz. The most prominent peak for the max-pool layer with dilation 2 is around 125Hz because $250/2 = 125$.

During the examination of the gradient peak, we noticed substantial differences in performance among the different architectures. Especially those with smaller kernel sizes or dilation parameters in their max-pool layers seemed to perform significantly better. Therefore, we decided to do a thorough inspection of each of the networks' performances and visualise the networks' gradients on the full and filtered datasets. The two filtered datasets, the high-passed dataset (> 60 Hz) and the low-passed dataset (< 40 Hz) and their combinations (i. e. low-pass for training and high-passed for validation), were created to see how the networks are influenced by having access to information from only some frequencies. The datasets were obtained as described in Section 3.1.5.



(a)



(b)

Figure 4.1 The blue line represents the difference between the output of a max-pool layer when original signal was given on input and when original signal with added white noise was given on input. The max-pool layer in **(a)** has kernel size (3, 1) and dilation (3, 1) and in **(b)** it has kernel size (3, 1) and dilation (2, 1).

4.1.1 Performance

The performances of the different architectures can be seen in Figure 4.2 for velocity and Figure 4.3 for absolute velocity. In the following points, we summarise the findings on the different datasets.

- **Full training and validation:** Some of the networks significantly outperformed the original Deep4Net (vel_k3_d3_sbp0)[2] from Hammer et al., 2021 when both trained and validated on full data. The best performing network was the network where the max-pool layer had no influence, namely the one with max-pool layer kernel size 1. It achieved the best correlation coefficients for both velocity and absolute velocity.

- **Full training and low-pass validation:** Training the networks on the full dataset and validating them on the low-passed dataset (< 40 Hz) caused a slight performance decrease for all the networks. While the decrease in performance was statistically significant, there is almost invisible when looking at Figures 4.2 and 4.3.

  Moreover, in order to achieve a statistically significant decrease in performance, we had to use a Butterworth filter of order 15 instead of order 3, which was previously used in Hammer et al., 2021. The 3rd order filter caused no significant change in the correlation coefficients.

  A Butterworth filter gradually attenuates the frequencies above the cut-off frequency (40 Hz in this case). The higher the filter order, the steeper the attenuation is (Butterworth et al., 1930). Therefore, the fact that the performance decrease followed only after the stronger filter was employed suggests that the networks use some minimal information from frequencies above 40 Hz but not particularly high frequencies (those in the high-gamma band), which were attenuated by both the 3rd as well as 15th order filter (Figure 4.4). The overall importance of frequencies above 40 Hz seems to be low.

- **High-pass training and validation:** To see if the networks can use information from the high-gamma frequency band, the networks were trained and validated on the high-passed dataset (> 60 Hz). The above chance correlations in Figure 4.2 and Figure 4.3 show that the networks can use some information from the frequency bands above 60 Hz. Nevertheless, the correlation coefficients for velocity are all below 0.1, suggesting that the information about velocity in the high-gamma band is very limited. We can see that the correlation coefficients values for absolute velocity are higher

[2] See Table 3.2 for abbreviation explanation.

than for velocity, demonstrating that frequencies above 60 Hz are more informative for absolute velocity decoding. Still, even for absolute velocity, the medians of the correlation coefficients stay below 0.2, showing that the information encoded in the high frequencies is not very useful for decoding.

- **Full training and high-pass validation:** We trained the networks on the full dataset and validated them on the high-passed data to test, whether the networks, when having access to full data, are using information from the high-gamma frequency band. Figures 4.2 and 4.3 show that most of the networks perform significantly better than chance level decoding. Nevertheless, the networks without max-pool layers {variable}_k1 for both velocity and absolute velocity are at chance decoding level. For absolute velocity, also the absVel_k2_d3 network did not reach correlations significantly higher than chance level decoding. The correlation coefficients for the other networks are all below 0.1 for velocity and mostly below 0.2 for absolute velocity. From this, we can conclude that the networks, when given access to, utilise information primarily from the low end of the frequency spectrum.

- **Low-pass training and high-pass validation:** Training on low-passed data and validation on high-passed data was also important to further study how the networks operate. We wanted to find out if they can somehow transfer information between two completely separate datasets. Because the cut-off frequency for the low-passed data is 40 Hz and for the high-passed data 60 Hz with a very steep filter, there is no frequency overlap between the two sets. Therefore, it would be interesting but also rather surprising if, from modulation in the low frequencies (below 40 Hz), the network would learn to use information about modulations in the high frequencies (above 60 Hz). Nevertheless, as can be seen in Figure 4.2 and Figure 4.3, the networks were unable to transfer any information.

We continue our analysis by looking at the gradients of the various architectures. The analysis is described in Section 4.1.2.

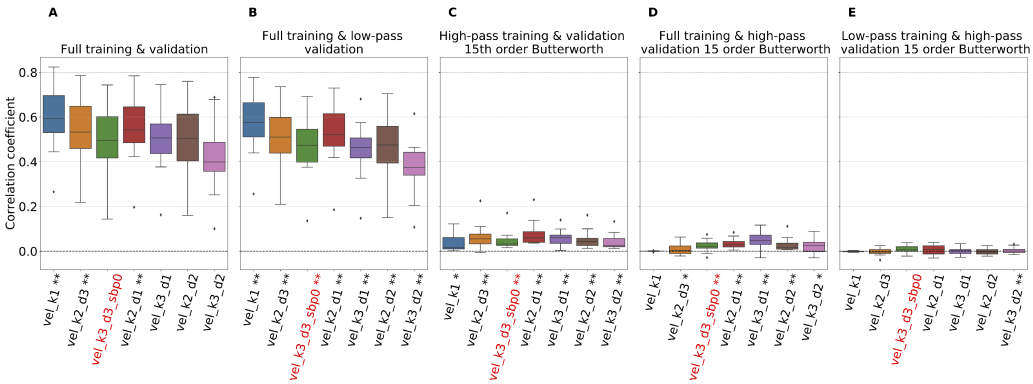


Figure 4.2 **Velocity** decoding correlation coefficients of the different CNNs established by architecture modifications. In all settings **A** - **E** the Deep4Net (vel_k3_d3_sbp0) from Hammer et al., 2021 is labeled red. **Graph A** shows the performance of the networks when trained and validated on the full dataset. The stars in this case denote performance significantly above the vel_k3_d3_sbp0. (** p <0.01), (* p < 0.05), Wilcoxon signed rank test. **Graph B** shows the correlation coefficients of the networks trained on full data and validated on low-passed data. The stars denote if the drop of performance was significant between this setting and setting **A**. (** p <0.01), (* p < 0.05), Wilcoxon signed rank test. **Graph C** shows the performance of the CNNs when trained and validated on high-passed data. **Graph D** shows performance when trained on full data and validated on high-passed data. **Graph E** shows performance when trained on low-passed data and validated on high-passed data. For **Graphs C** - **E** the stars denote above chance performance - (** : p <0.01), (* : p < 0.05), Wilcoxon signed rank test.

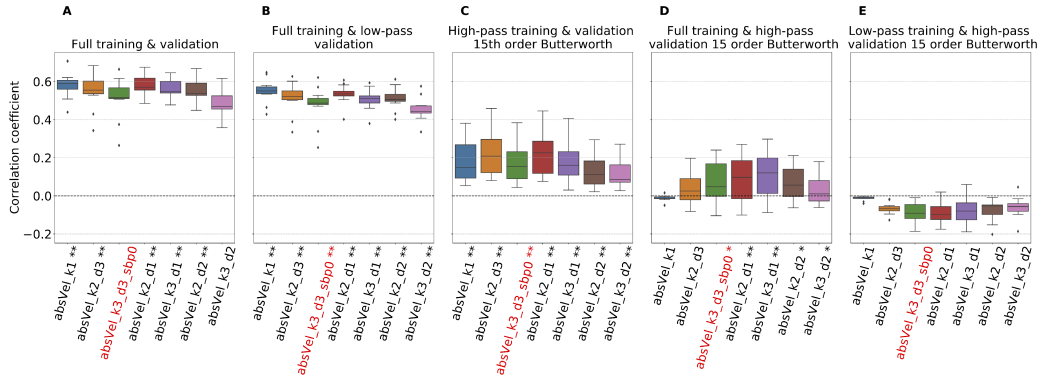
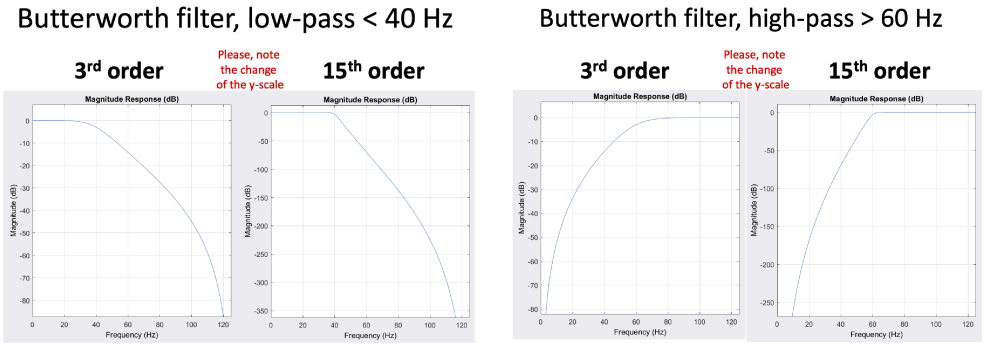


Figure 4.3 **Absolute velocity** decoding correlation coefficients of the different CNNs established by architecture modifications. In all settings **A** - **E** the Deep4Net (absVel_k3_d3_sbp0) from Hammer et al., 2021 is labeled red. **Graph A** shows the performance of the networks when trained and validated on the full dataset. The stars in this case denote performance significantly above the absVel_k3_d3_sbp0. (** p <0.01), (* p < 0.05), Wilcoxon signed rank test. **Graph B** shows the correlation coefficients of the networks trained on full data and validated on low-passed data. The stars denote if the drop of performance was significant between this setting and setting **A**. (** p <0.01), (* p < 0.05), Wilcoxon signed rank test. **Graph C** shows the performance of the CNNs when trained and validated on high-passed data. **Graph D** shows performance when trained on full data and validated on high-passed data. **Graph E** shows performance when trained on low-passed data and validated on high-passed data. For **Graphs C** - **E** the stars denote above chance performance - (** : p <0.01), (* : p < 0.05), Wilcoxon signed rank test.



(a) Comparison between 3rd and 15th order low-pass Butterworth filter, cutoff frequency 40 Hz

(b) Comparison between 3rd and 15th order high-pass Butterworth filter, cutoff frequency 60 Hz

Figure 4.4

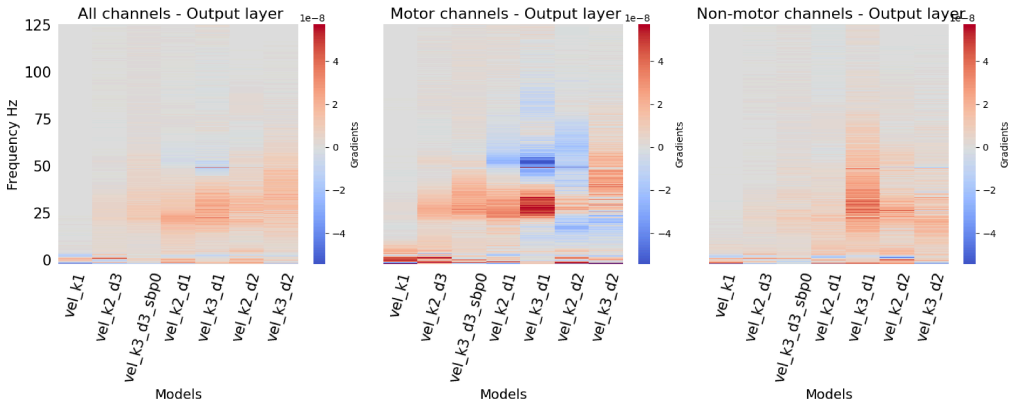
4.1.2 Gradients

The differences in performance among the networks reinforce the interest in the gradients of the various architectures. Since some of the networks perform significantly better than the initial Deep4Net, we analyse gradients of the different architectures to see if the reason for better performance is their ability to use information from the high-gamma band. We perform the gradient visualisation of all the architectures with max-pool kernel sizes 1, 2 and 3 and dilations as powers of 1, 2 and 3. See Table 3.2 for a detailed description of the architectures, and Section 3.2.5 for a description of the visualization technique.

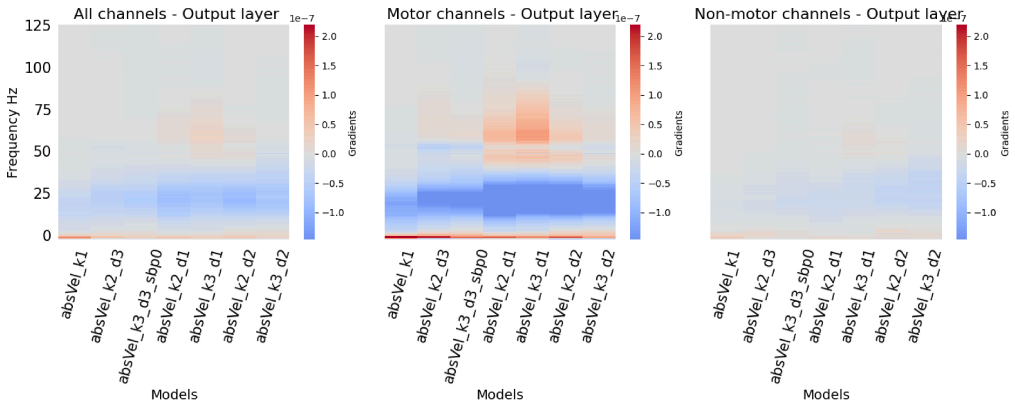
The results show differences in gradients among the architectures. Gradients of all intermediate layers can be found in Appendix A. Figure 4.5 depicts the gradients of the output layer of each of the inspected architectures.

Based on the performances of the networks presented in Section 4.1.1 and the gradients presented in this section, these important and interesting observations can be made:

- The networks focus mostly on motor channels when making predictions. This is to be expected when they are tasked with decoding movement.
- There are obvious differences between the gradients for velocity and absolute velocity. Nevertheless, networks with interest in wider frequency ranges for velocity decoding have interest in wider frequency ranges for absolute velocity decoding and vice-versa.
- For both variables, the network without max-pool, denoted as {variable}_k1 is the best performing architecture. And in both cases, it is also the network that is most interested in modulations in the low-frequency bands. This suggests that using the information in the high-gamma frequency band is not necessarily advantageous.
- The networks which exhibit higher interest in information from the higher frequencies, namely the {variable}_k2_d1, {variable}_k3_d1 and {variable}_k2_d2 are also those which can perform significantly above chance when trained on full data and validated on high-passed data for both velocity and absolute velocity. This suggests consistency between the gradient visualisation and the performance analysis.



(a)



(b)

Figure 4.5 Gradients of the different CNN architectures decoding **(a) Velocity** and **(b) Absolute velocity**. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4

4.2 Shifting the predicted time-point

We described in Section 3.2.3 that the receptive field is non-uniform - it considers mostly input-points in its centre; and that in the original, non-shifted setting (causal prediction), the predicted time-point is located just outside the receptive field, which places it relatively far from the centre of the receptive field.

In Section 4.1.1 we discovered differences in performance among networks. We also noticed that better performance seemingly correlates with a smaller receptive field. To visualise this, we created Figure 4.7 where we can observe a clear descending pattern of performance when increasing the size of the receptive field. The only exceptional network is the k3_d3 network which performs well with a relatively large receptive field.

The descending pattern and the non-uniformity of the receptive field corroborate the hypothesis that the distance of the predicted time-point to the centre of the receptive field plays an important role in the prediction power of the architectures. A smaller size of the receptive field means a smaller distance between the predicted time-point and receptive field centre, and therefore, possibly access to more relevant signals recorded closer to the predicted movement execution.

To study this further, we shift the inputs and predictions so that the iEEG signal, which was recorded at the same time as the predicted movement was executed, is in the centre of the receptive field[3]. An illustration of this is in Figure 4.6. We present how this shift affects the performances and gradients of the various architectures in Section 4.2.1.

Besides shifting the predicted time point in one big step to the centre of the receptive field, we also tried shifting the predicted time-point in smaller steps across the receptive field. This analysis which is introduced in Section 4.2.3 was performed only on the original Deep4Net which we denote as {variable}_k3_d3_sbp0. It compares how the performance and gradients of the network change when the predicted time-point is shifted in small steps across the receptive field, using a changing ratio of information from the future and from the past.

4.2.1 Shifting the predicted time-point to the centre of the receptive field

In this section, we look at how the shift of the predicted time-point into the centre of the receptive field influences the performance and gradients of the different network architectures.

[3]This causes the procedure to be unsuitable for online BCI because half of the input window uses information from the future.

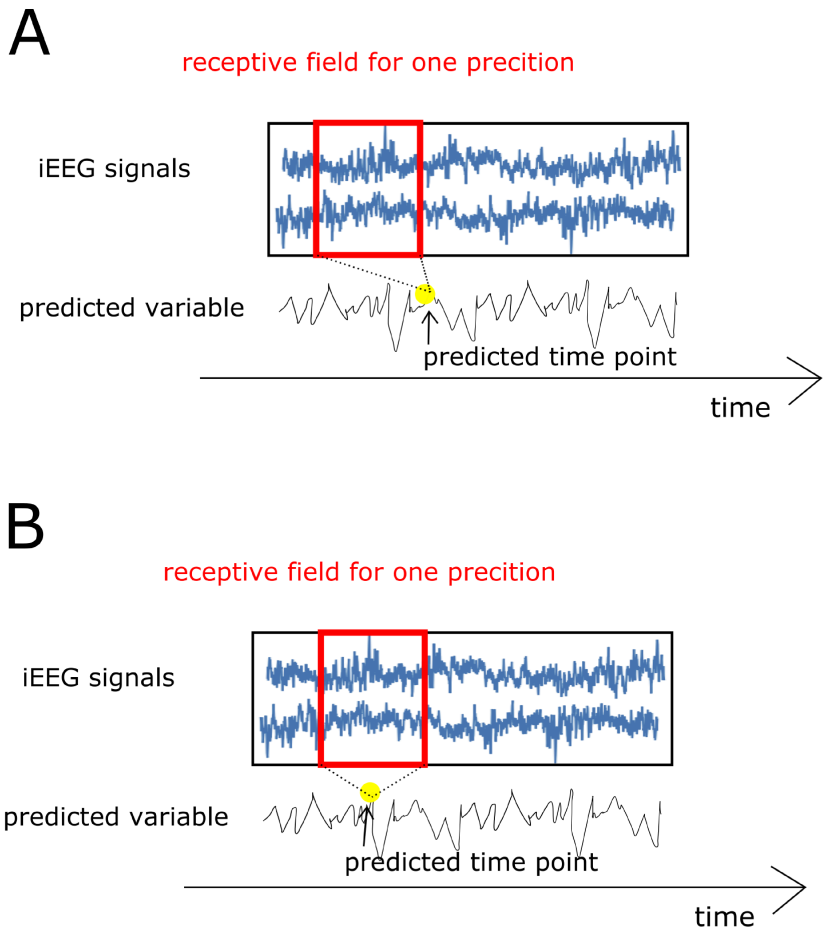


Figure 4.6 This Figure shows the difference between the original non-shifted setting (causal prediction) - **Graph A** and the shifted setting - **Graph B**. In Graph **A** we can see that the receptive field used for predicting the yellow time-point is recorded prior to the predicted time-point. This allows for causal prediction and usability in online BCI. In Graph **B** the predicted time-point is in the middle of the receptive field. Half of the signals used for its prediction was recorded prior to it, and the other half was recorded later than the movement was executed.

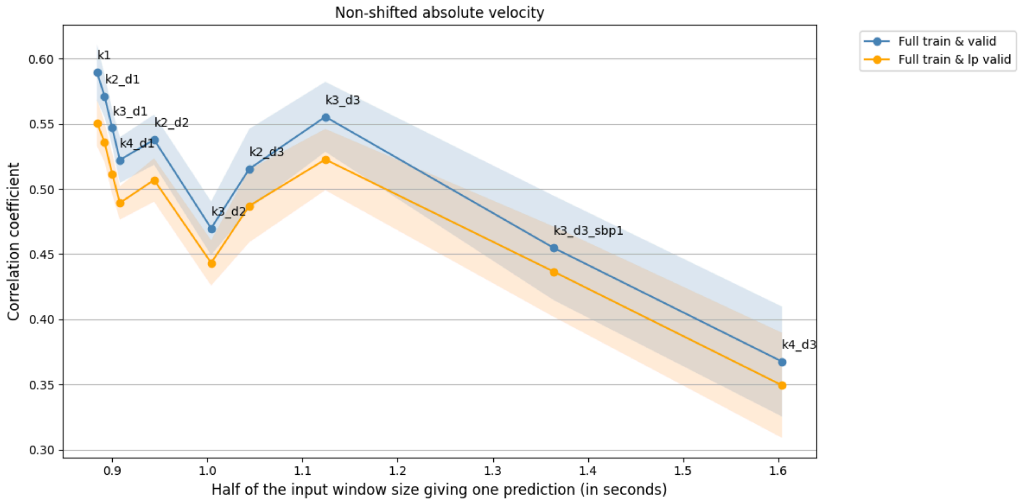


Figure 4.7 The average performance of the different CNN architectures with respect to the size of their receptive field. Blue: Full training and validation CC; Light-blue: full training and validation CC standard deviation; Orange: Full training and low-pass validation CC; Light-orange: Full training and low-pass validation CC standard deviation

Performance

The performance related results are displayed in Figure 4.8 and Figure 4.9. We summarise how the shift influences performances on the different datasets in the following points:

- **Full training and validation:** It is obvious that the shift greatly improves the performance of all the networks on the original full data training and validation and that in this setting, the performance differences between the architectures are diminished.

- **Full training and low-pass validation** After the shift, the performance of the networks in the shifted setting drops significantly for all the networks as it does in the non-shifted original setting. Nevertheless, even though we found a statistically significant drop, visually, the difference between the correlation coefficient of networks trained and validated on the full data and the networks trained on full data and validated on low-passed data seems negligible. The negligible difference suggests that the networks use some information from frequencies above 40 Hz in the shifted setting, but similarly to the original setting, only in a very limited manner.

- **High-pass training and validation** The performance of the networks on the high-gamma dataset increases with the shift, especially for absolute

velocity. This suggests that the modulations in the frequencies above 60 Hz become more informative with the shift.

- **Full training and high-pass validation** After the shift, the performance of the networks trained on full data and validated on high-pass data decreased. For absolute velocity and velocity, fewer networks performed above chance level and those that stayed above chance level often perform above chance level less significantly. This suggests that the networks, when having access to full data, depend on the information from the high frequencies less than in the original non-shifted setting.

Two things can cause the overall improvement in performance:

1. By the network being able to focus on signals recorded directly before the movement execution.

2. By the network having access to information from the future.

We hypothesise that it is most likely a combination of the two. Conclusion about how much each of the above-described possibilities influences the prediction improvement cannot be made from the presented experiments. It would be interesting to build a network with a uniform receptive field and then conduct experiments that would clarify this.

If the performance improvement was caused solely or mainly by access to information directly preceding the predicted time-point, a network with a uniform receptive field (one which allows padding) could potentially bring an improvement, similar to the one we achieved by shifting the predicted time-point, while retaining its usability for online BCI. Its uniform receptive field would allow focusing on information directly preceding the movement without also considering information from the future. Nevertheless, such an analysis is out of the scope of this thesis.

Gradients

We theorised that the information about the velocity and absolute velocity of the movement could be encoded in the high-gamma frequencies of iEEG only in the moments directly preceding the movement. Therefore, the networks were unable to use it in the original non-shifted setting because they were biased towards signals recorded too far in the past (see Section 3.2.3). The drop of performance of the networks after the shift, when trained on full data and validated on high-passed data, visible in Figures 4.8 and 4.9 speaks against this theory. To further test this hypothesis, we compared the gradients between the networks in the original non-shifted setting (causal prediction) and the shifted setting where

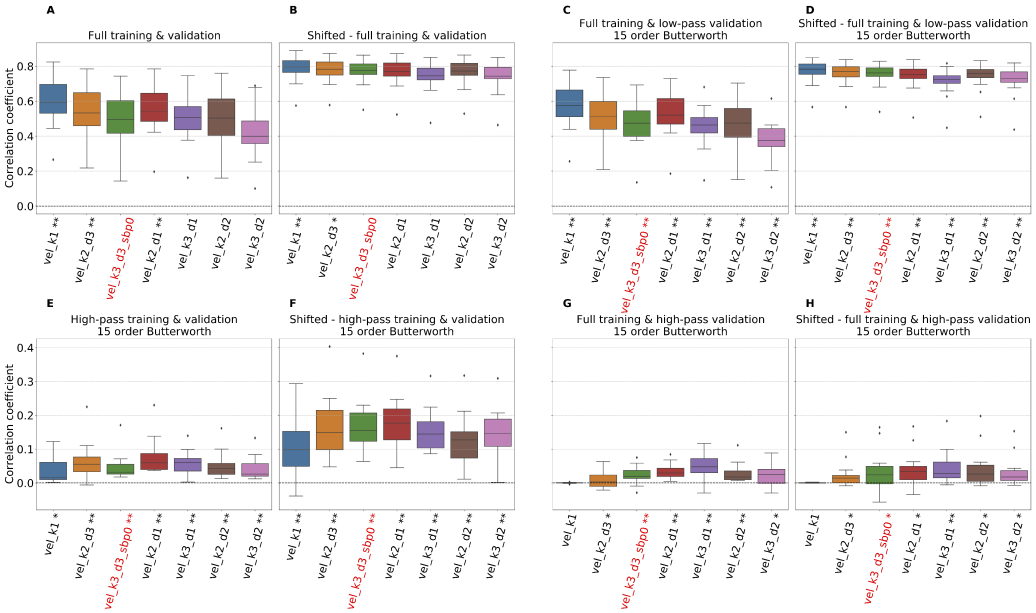


Figure 4.8 **Velocity** decoding correlation coefficients comparison between original causal prediction **Graphs A, C, E, G** and the shifted (acausal) prediction **Graphs B, D, F, H**. In all settings **A - E** the Deep4Net (k3_d3_sbp0) from Hammer et al., 2021 is labeled red.

Graphs A and **B** compare the performance of the networks when trained and validated on the full dataset. A The stars in **A** and **B** denote performance significantly above the Deep4Net in the same setting. (** p <0.01), (* p < 0.05), Wilcoxon signed rank test.

Graphs C and **D** show the correlation coefficients of the networks trained on full data and validated on low-passed data. The stars in **C** denote if the CCs are significantly lower compared to **A**. The stars in **D** denote if the CCs are significantly lower compared to **B**. (** p <0.01), (* p < 0.05), Wilcoxon signed rank test.

Graphs E and **F** compare the CCs of the CNNs when trained and validated on high-passed data. **Graphs G** and **H** compare performance when trained on full data and validated on high-passed data. For **Graphs E - H** the stars denote above chance performance - (** : p <0.01), (* : p < 0.05), Wilcoxon signed rank test.

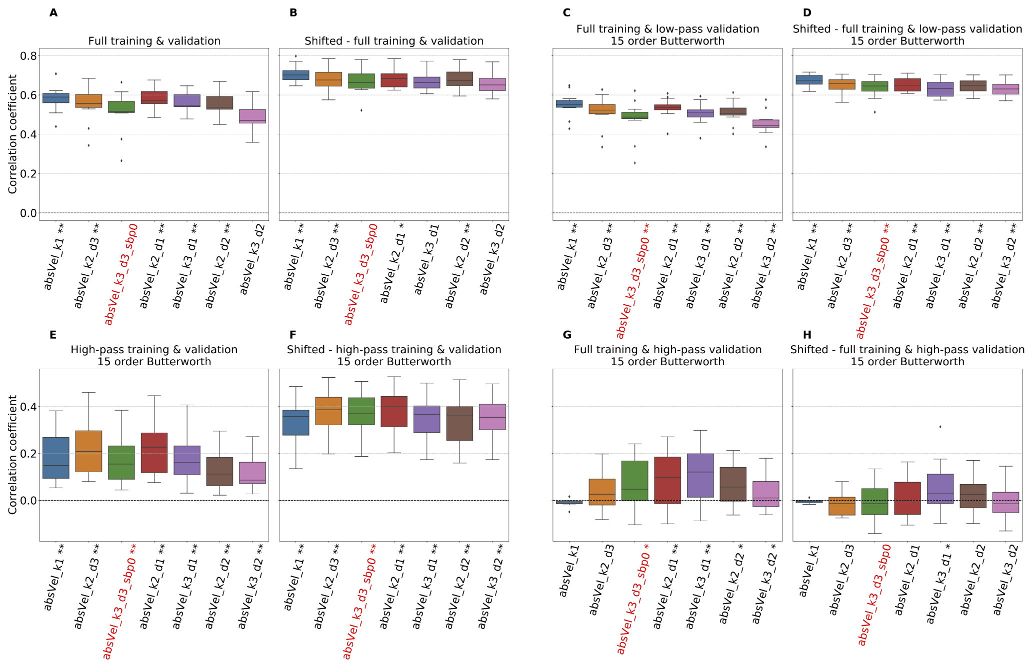


Figure 4.9 **Absolute velocity** decoding correlation coefficients comparison between original causal prediction **Graphs A, C, E, G** and the shifted (acausal) prediction **Graphs B, D, F, H**. In all settings **A - E** the Deep4Net (k3_d3_sbp0) from Hammer et al., 2021 is labeled red.

Graphs A and **B** compare the performance of the networks when trained and validated on the full dataset. A The stars in **A** and **B** denote performance significantly above the Deep4Net in the same setting. (** p <0.01), (* p < 0.05), Wilcoxon signed rank test.

Graphs C and **D** show the correlation coefficients of the networks trained on full data and validated on low-passed data. The stars in **C** denote if the CCs are significantly lower compared to **A**. The stars in **D** denote if the CCs are significantly lower compared to **B**. (** p <0.01), (* p < 0.05), Wilcoxon signed rank test.

Graphs E and **F** compare the CCs of the CNNs when trained and validated on high-passed data. **Graphs G** and **H** compare performance when trained on full data and validated on high-passed data. For **Graphs E - H** the stars denote above chance performance - (** : p <0.01), (* : p < 0.05), Wilcoxon signed rank test.

the predicted time point is in the centre of the receptive field (acausal prediction). This analysis was performed for all architectures for all their intermediate convolutional layers[4] on 1. the full dataset and 2. the high-passed dataset. The complete results can be found in Appendix B. Below, we show results from the convolutional layer in the third convolutional block (conv_3), which represents the overall trend.

1. Gradients of networks which are **trained and validated on full data** are displayed in Figures 4.10 (velocity) and 4.11 (absolute velocity). We observe that with the shift the networks across all architecture seem to refine their focus to more narrow frequency bands. This can be illustrated on for example Figure 4.10a where in the original, non-shifted setting, the vel_k1 network has high-gradient values for frequencies up to 25 Hz when looking at motor channels. In the shifted setting Figure 4.10b the band with high gradient values of the same network vel_k1 narrows to frequencies closer to 0.

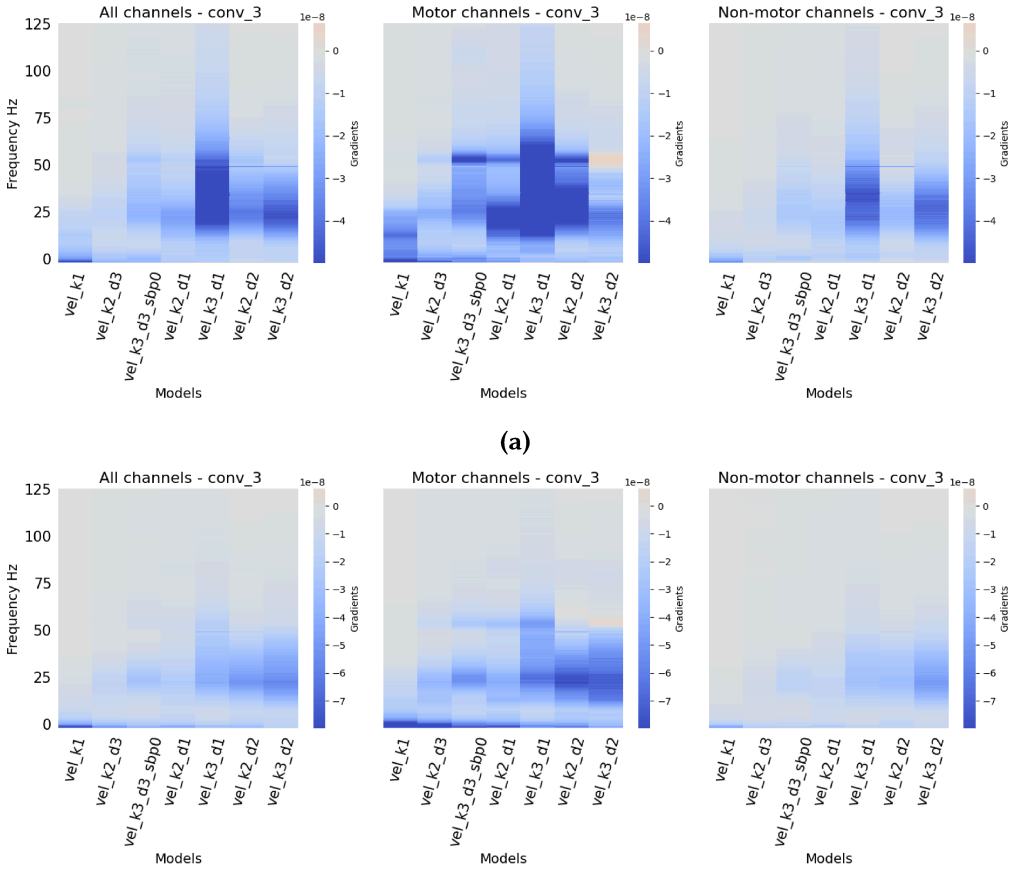
   The shift did not cause an increased utilisation of information from the high-gamma frequency band for any of the networks. Rather it seems, that the shift allowed access to less noisy information in the clearly informative bands such as the alpha (4 - 12 Hz) and beta (13 - 30 Hz) bands, and the network did not compensate with information from other frequencies.

2. Networks which were **trained and validated on high-passed data** can be found in Figures 4.12 (velocity) and 4.13 (absolute velocity). Again, we only chose to display gradients of one layer to illustrate a behaviour shared by all layers. The gradients of the remaining layers can be found in Appendix B. What we observe in the gradients of the networks trained on high-passed datasets is different (even opposite) from what we observe in gradients of networks trained on the full dataset. The networks trained on high-passed data in the shifted setting exhibit interest in the same or sometimes a broader range of frequencies above the 60 Hz cut-off frequency than the networks trained on high-passed data in the original, non-shifted setting. For the velocity gradients in Figure 4.12 the frequency ranges broaden, especially for the vel_k3_d3_sbp0 model and generally, the gradients become higher for non-motor channels. In Figure 4.13 for absolute velocity, we do not see a broadening of interesting frequency bands for any network for any channels. Instead, they stay the same.

   The increase in performance on high-passed datasets suggests that indeed the information in high-frequency bands in signals from the future

[4] omitting the temporal and spatial convolution from the first block

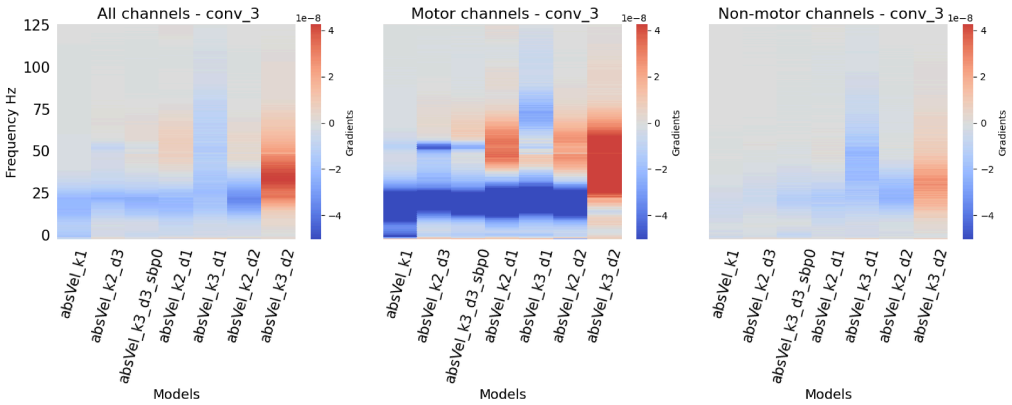
or directly preceding the movement contains more information about the movement. The broadening of interesting frequency ranges above 60 Hz for velocity also corroborates it, at least for this variable. However, the fact that the networks trained in the shifted setting on full data do not increasingly use high-gamma, while their performance increases significantly compared to the non-shifted setting, points to the information in high-gamma being informative but redundant when having access to information from all frequencies.



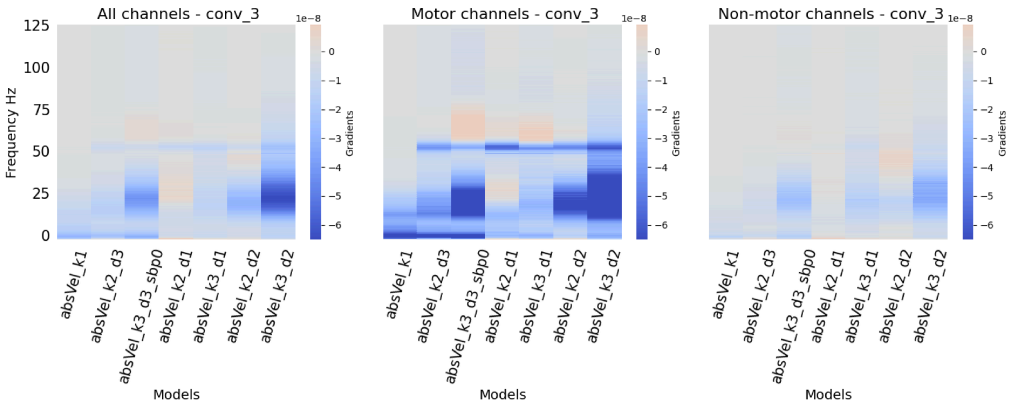
(a)

(b)

Figure 4.10 Gradients of the different CNN architectures trained to decode velocity in **(a)** the original setting (causal prediction) and **(b)** in the shifted setting (acausal prediction). Full data was used for both training and validation.

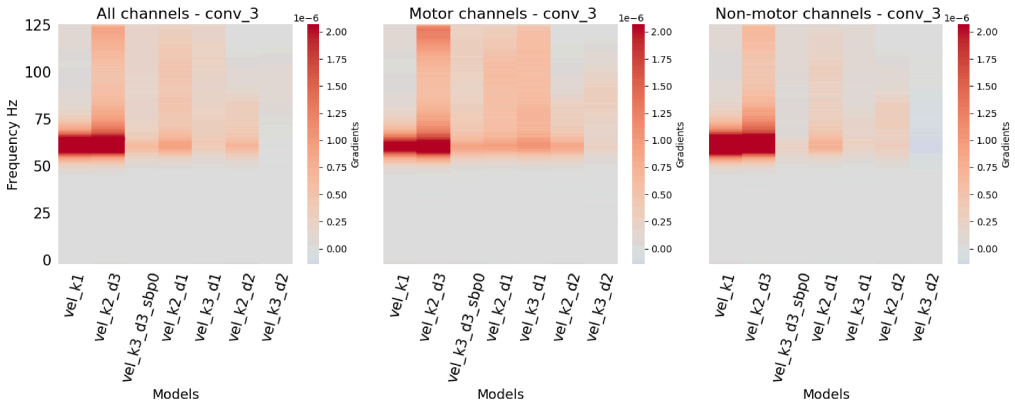


(a)

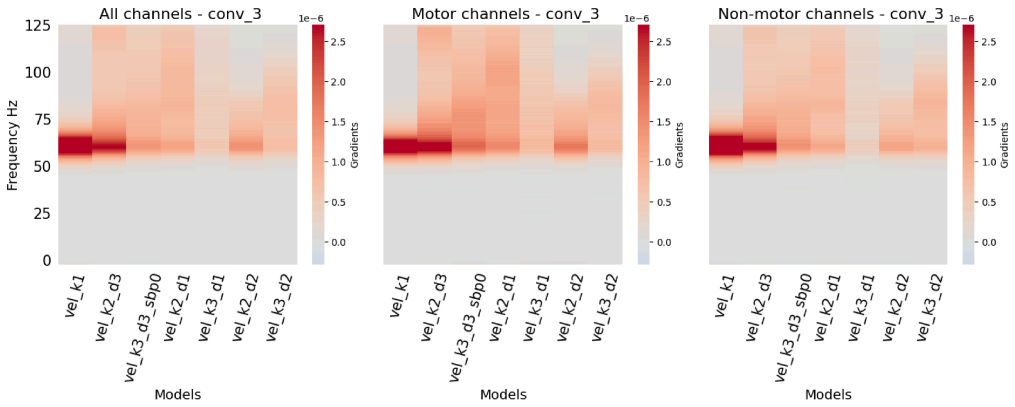


(b)

Figure 4.11 Gradients of the different CNN architectures trained to decode absolute velocity in **(a)** the original setting (causal prediction) and **(b)** in the shifted setting (acausal prediction). Full data was used for both training and validation.

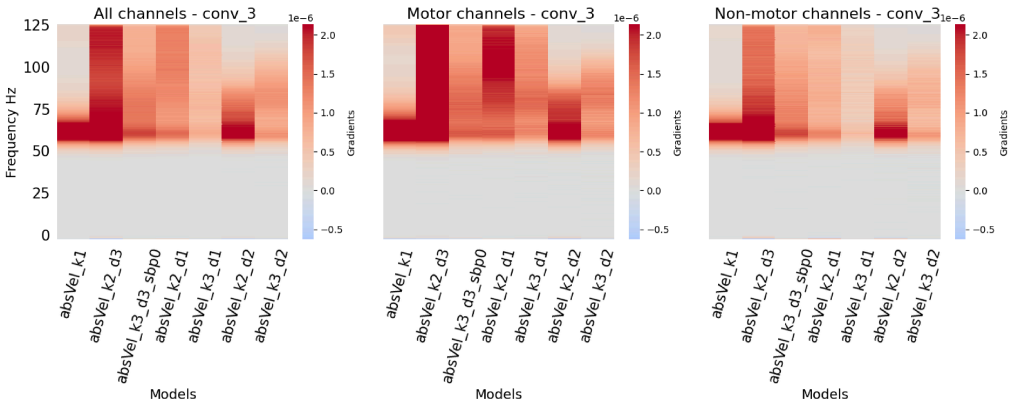


(a)

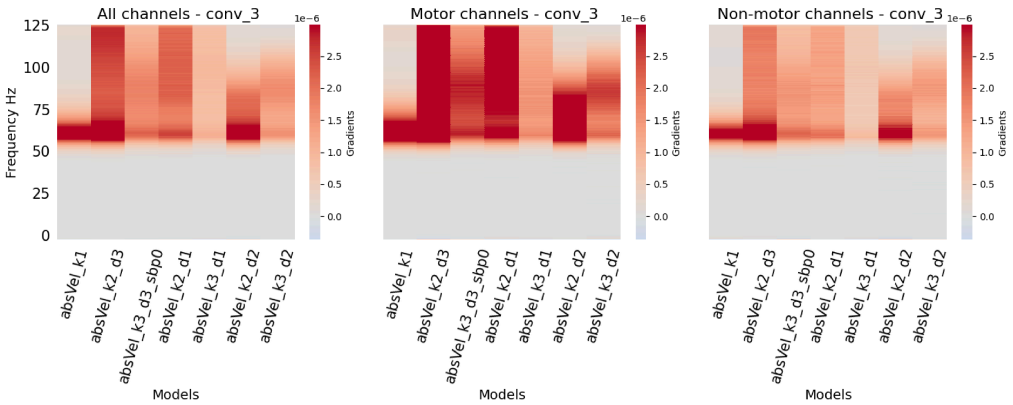


(b)

Figure 4.12 Gradients of the different CNN architectures trained to decode velocity in **(a)** the original setting (causal prediction) and **(b)** in the shifted setting (acausal prediction). High-passed data was used for both training and validation.



(a)



(b)

Figure 4.13 Gradients of the different CNN architectures trained to decode absolute velocity in **(a)** the original setting (causal prediction) and **(b)** in the shifted setting (acausal prediction). High-passed data was used for both training and validation.

Summary

The following observation can be made when looking at the performances and gradients of the various CNN architectures in the shifted vs non-shifted (original) setting.

- The shift improves the performance of networks 1. trained and validated on the full dataset; 2. trained on the full dataset and validated on low-passed data; 3. trained and validated on high-passed data. This is true for both velocity and absolute velocity.
- The shift attenuates the differences in performance between the different CNNs for the full training and validation. When looking at graphs **B** in both Figure 4.8 and Figure 4.9, we can see that the number of network which have a significantly better performance than the original Deep4Net (k3_d3_sbp0) decreases compared to **A**.
- The shift does not improve performance for the CNNs trained on full data and validated on high-pass data (graphs **G** and **H** in both velocity - Figure 4.8 and absolute velocity - Figure 4.9). This result, together with the more narrow frequency bands the networks focus on after the shift (Figures 4.10 and 4.11), show that the networks do not start focusing on high-gamma with the shift when having access to the whole frequency spectrum. Rather the opposite. The networks more clearly refine their focus, often on information from the lower frequency bands. Modulations in these low-frequency bands become more informative with the shift; therefore, the performance increases and the interest of the CNNs in higher frequencies drops.
- The modulations in the high-gamma band become more informative for decoding with the shift, thus the increase in performance when training and validating on the high-passed dataset. Nevertheless, as we state in the point above, not even this motivates the networks to use information from the high-gamma band when having access to information from lower frequencies.
- It is true in the shifted setting, as was in the non-shifted setting, that the network without max-pool (k1), which is almost solely focused on low-frequency modulations, performs the best and significantly above the original Deep4Net (k3_d3_sbp0).

From the observations above, we can state that the modulations in the high-gamma band are not particularly informative for velocity and absolute velocity decoding. They contain information the CNNs are able to use for decoding.

Nevertheless, it is not an advantage for the CNN to use high-gamma modulations when having access to all frequencies; rather, it harms its performance. It is better if the CNNs focus on low frequencies.

4.2.2 Decoding absolute velocity as the absolute value of velocity

We can see in Figures 4.8 and 4.9 that when training and validating on the full data in shifted setting (Graphs B), the velocity correlation coefficients are higher than the absolute velocity correlation coefficients. This seems a bit counter-intuitive because absolute velocity can be derived from velocity only as the absolute value of velocity. Therefore, it could be at least as well decoded. To see if absolute velocity can be decoded better when taking absolute values of velocity, we trained a network on velocity data and validated it on absolute values of the predictions and absolute velocity data. The results show that the correlation coefficients are significantly worse than when training the network on absolute velocity data (Figure 4.14). Even though it seems that it might be better to learn to decode absolute velocity as velocity and then take the absolute value, we find that it is not.

This might come as surprising but can be explained mathematically. The correlation coefficient is derived from the mean and standard deviation of the variables as:

$$CC_{(X,Y)} = \frac{E[(X - \mu_X)(Y - \mu_Y)]}{\sigma_x \sigma_y} \tag{4.1}$$

where X and Y are two random variables σ_x, σ_y are the standard deviations of X and Y, μ_x, μ_y are the means of X and Y.

No equitation tells us how absolute value changes the mean and standard deviation of an arbitrary variable. Therefore, the means and standard deviations of the predicted values and the gold values change independently when taking their absolute values and the correlation coefficient changes. It is not equivalent to take the correlation coefficients of two variables and the correlation of absolute values of two variables. To show this non-equivalency, let X be a random variable with values $X = \{1, 0, 1, 0, 1\}$ and Y be a random variable with values $Y = \{2, 1, -2, 1, 2\}$. In this case $CC_{(X,Y)} = -0.1111$ and $CC_{(|X|,|Y|)} = 1$.

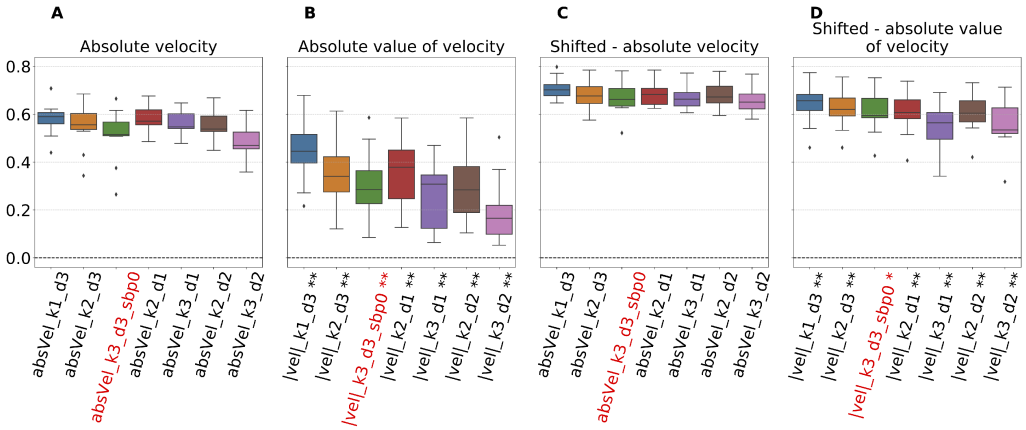


Figure 4.14 Absolute velocity decoding from a absolute velocity data **Graphs A, C** and absolute velocity decoding when taking absolute values of networks trained on velocity data **Graphs B, D**. In all settings **A - D** the Deep4Net (k3_d3_sbp0) from Hammer et al., 2021 is labeled red.

Graphs A and **B** compare the performance of the networks when trained and validated on the full dataset. A The stars in **B** denote performance significantly below the same architecture in graph **A**. (** p <0.01), (* p < 0.05), Wilcoxon signed rank test.

Graphs C and **D** compare the performance of the networks when trained and validated on the full dataset in the shifted setting. A The stars in **D** denote performance significantly below the same architecture in graph **C**. (** p <0.01), (* p < 0.05), Wilcoxon signed rank test.

4.2.3 Shifting the predicted time-point across receptive field

Besides the big shift of the predicted time-point from the edge of the receptive field to the centre, we also studied what happens if we shift the predicted time-point in small steps across the receptive field. The shifts range from -1 second to 1 second from the centre of the receptive field, which we denote as 0. When the predicted time-point is 0, we use precisely half the information from the past and half the information from the future. When shifting towards the negative values, we use more information recorded after the execution of the predicted movement (information from the future). Vice-versa, when shifting towards positive values, we increasingly use information recorded prior to the predicted movement coming closer to the original non-shifted setting. The step size is 0.1 s which is equivalent to 25 samples. This experiment allows us to observe how the shifting gradually influences the performance and gradients of the network.

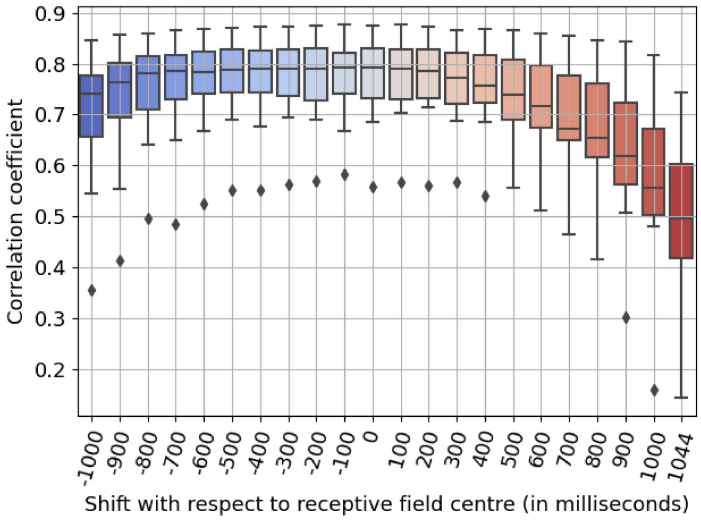
We chose to perform this analysis only on one architecture, namely, the original Deep4Net (k3_d3_sbp0).

Performance

How the performance changes with the gradual shifting of the predicted time-point is displayed in Figure 4.15. We can observe the gradual decrease in performance when increasing the distance of the predicted time-point to the receptive field centre in both directions. This is to be expected. Similarly to performance in Section 4.2.1 we do not know if the fact that the performance peaks in the centre of the receptive field is due to having information directly preceding the movement or having access to information from the future. Interestingly, the decoding performance drops slower when using information predominantly from the future than when using information predominantly from the past. This could indicate that the information about movement in iEEG signals is dominated by the representation of movement execution rather than its planning/preparation. To properly investigate the informativeness of past vs future signals, however, we would need to have a network with a uniform receptive field.

Gradients

How the gradients are changing with the gradual shifting is visualized in Figure 4.16. We again chose to display one layer (the output layer) in the main text to represent the overall trend. The gradients of the remaining layers for both the full dataset and the high-passed data can be found in Appendix C. In the graph for absolute velocity (Figure 4.16b), we can observe the trend of broadening the



(a)

(b)

Figure 4.15 The boxplots in **(a)** show how the CCs of the original Deep4Net (vel_k3_d3_sbp0) changes for velocity decoding when shifting the predicted time-point across the receptive field. The boxplots in **(b)** show the same but for absolute velocity. Zero milliseconds on the x-axis represent the predicted time-point being shifted to the centre of the receptive field. The network uses less information from the future, and the predictions become more causal when moving to the right. The 1041 ms mark on the x-axis is equivalent to the original fully causal prediction as described in Hammer et al., 2021. When moving from 0 to the left, the network uses more and more information from the future.

frequency ranges with high gradient values when shifting the predicted time-point from the centre of the receptive field to the right towards positive values and more causal prediction, which is closer to the original setting. This is what we expected because it corresponds to the results from Section 4.2.1 where the frequency bands with high gradient values are more narrow in the shifted setting (predicted time-point in the centre of the receptive field) and wider when performing causal prediction. Interestingly, the frequency bands in which the network is interested widen also when shifting the predicted time-point toward the negative values (using more information from the future).

When plotting the same graph for velocity Figure 4.16a, we do not observe this behaviour so clearly. Instead, there is a periodicity in the positive and negative value of the gradients for the different shift values. It is unclear why this periodicity occurs.

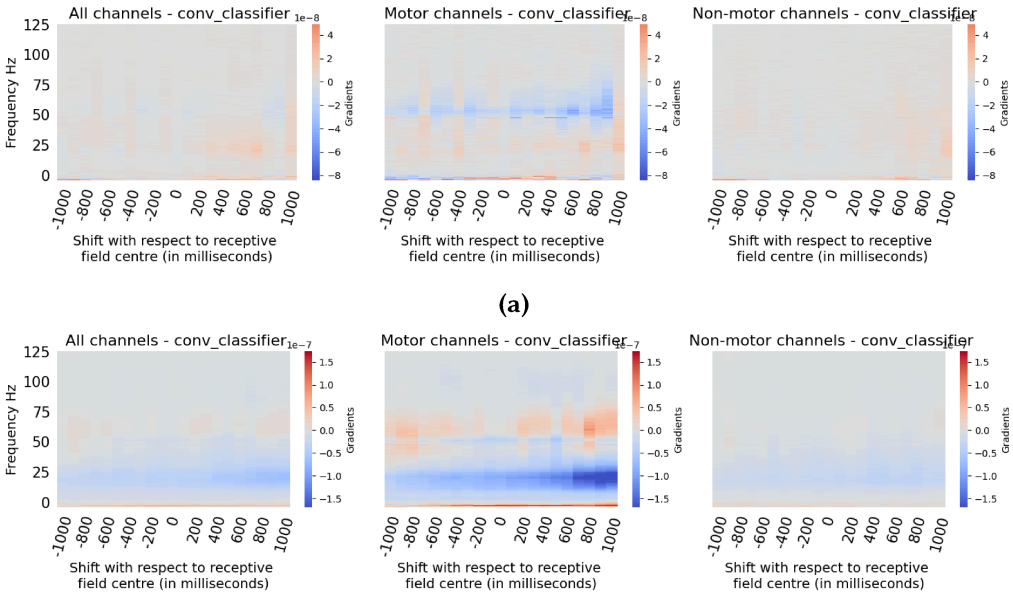


Figure 4.16 The graph **(a)** shows the changes in gradients of the original Deep4Net (vel_k3_d3_sbp0) trained to decode velocity when shifting the predicted time-point across the receptive field. The graph **(b)** show the same but for absolute velocity. Zero milliseconds on the x-axis represent the predicted time-point being shifted to the centre of the receptive field. The network uses less information from the future, and the predictions become more causal when moving to the right. The 1041 ms mark on the x-axis is equivalent to the original fully causal prediction as described in Hammer et al., 2021. When moving from 0 to the left, the network uses more and more information from the future.

Summary

We have further confirmed the conclusions we drew previously in Section 4.2.1. Indeed, the network focuses on more narrow frequency bands when given better access to information close to the predicted time-point. This also means lower interest in the high-gamma frequency band when achieving better performance. It corroborates what we have established so far about the information in the high-gamma band being inferior for velocity and absolute velocity decoding compared to information from lower frequency bands.

4.3 Spectral whitening

How the networks react to datasets, which were whitened (i. e., the amplitudes of all frequencies were normalised to 1 - see Section 3.1.5) was one of our interests because when we look at the spectrum of the original signal (see Figure 3.2), the amplitudes of the frequencies decrease exponentially with increasing frequency. This decrease is common in biological signals and could be why the CNNs ignore high-frequencies when making predictions. This section evaluates the performances and visualises gradients of all the architectures on the whitened datasets and compares the results to performances and gradients from the original non-whitened settings.

4.3.1 Performance

We carry out the whitening on full as well as high-passed (> 60 Hz) and low-passed datasets (< 40 Hz) and their combinations. Figure 4.17 and Figure 4.18 show how the predictions change compared to predictions on non-whitened signals. We summarize the behavior the different datasets in the points below:

- **Full training and validation:** It is obvious that for the full training and validation for both velocity (Figure 4.17 graphs **A** and **B**) and absolute velocity (Figure 4.18 graphs **A** and **B**), the correlation coefficient of all networks dropped significantly. The network experiencing the lowest drop of performance due to spectral whitening was the network without max-pool ({variable}_k1).

- **Full training and low-pass validation:** The drop of performance between full training and validation and full training and low-pass validation, was bigger with whitening (graphs **C** and **D** in Figures 4.17 and 4.18) compared to the original non-shifted setting. This suggests that spectral whitening increased the use of the high-gamma frequency when having access to all frequencies.

- **High-pass training and validation:** When looking at graphs **E** and **F** in both Figure 4.17 and Figure 4.18 we see that in the case of the high-passed datasets the performance did not increase or decrease significantly for any of the networks. This shows that the low amplitudes of these frequencies are not the issue when decoding from them because increasing it does not aid the network in making predictions from them. Neither does increasing it harm them.

- **Full training and high-pass validation:**This was the only scenario where spectral whitening helped the CNNs to achieve better CCs. In the

case of velocity Figure 4.17 graphs **G** and **H** a statistically significant increase compared to the same non-whitened setting was only for the vel_k1 network. In the case of absolute velocity Figure 4.18 graphs **G** and **H** five out of the seven architectures experienced a statistically significant increase in performance due to spectral whitening. The CCs increases after spectral whitening in this last scenario show that at least some networks indeed learned to use more high-gamma when having access to full data.

4.3.2 Gradients

When looking at the gradients of the networks trained on whitened data in Figure 4.19[5], we observe that the networks indeed use modulations in the high-gamma frequency bands for their predictions. This finding is also supported by the fact that the networks trained on full data and validated on high-passed data experienced an increase in the correlations coefficients with spectral whitening. For absolute velocity, the gradients have high values in bands across almost the whole frequency range. Interestingly, for velocity, instead of having more uniform gradients across all frequencies, they partially invert their focus from low frequencies to high frequencies.

Among CNNs decoding velocity, the only architecture that stays mostly focused on frequencies below 25 Hz is the vel_k1 CNN - the network without max-pool layers. The same is true for absolute velocity, where the network without max-pools (absVel_k1) also stays mostly focused on the low frequencies even though a slight increase in gradient values for frequencies around 75 Hz can be observed. We can combine this information about the gradients with the performance graphs in Figure 4.19, where we can see that the networks without max-pools, which were least influenced by the spectral whitening in terms of focusing on higher frequencies, are those for which the CCs dropped least. This fact suggests that using information from the high-gamma frequency is possible, but it does not seem to help achieve better correlation coefficients. On the contrary, using information from the high-gamma frequency band is not helpful for better network performance and using information in the low frequencies leads to better prediction power even on the whitened datasets.

Overall spectral whitening forced most of the networks into using high-gamma. At the same time, it harmed the prediction power of the networks. Therefore, we can state that while the information in the high-gamma frequency band of iEEG signals holds information about velocity and especially absolute velocity, the networks perform better when they focus on information

[5] We present the gradients only for the output layer to highlight our observations. The results for all layers can be found in Appendix D.

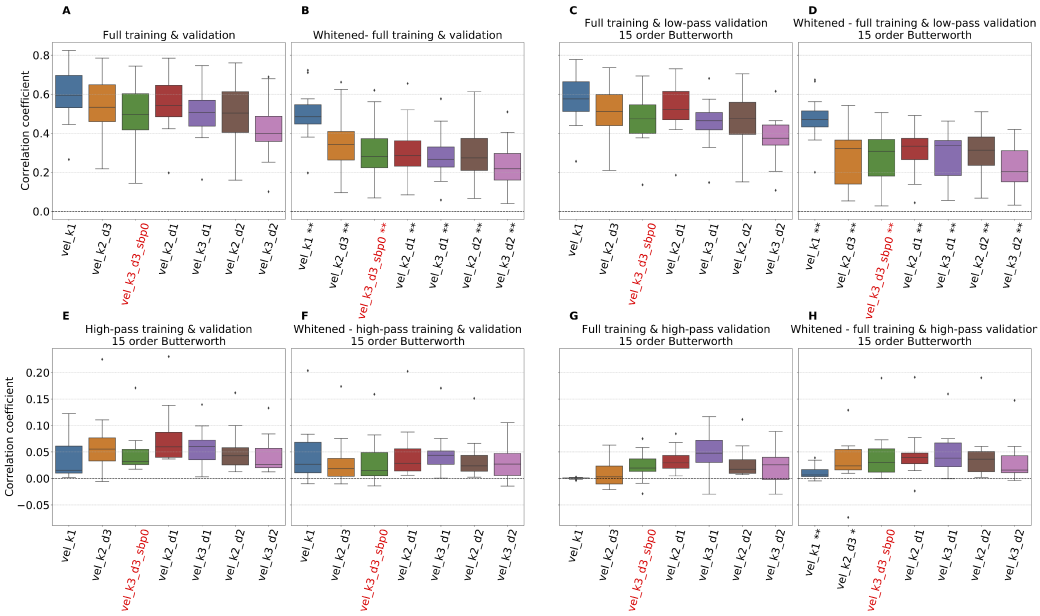


Figure 4.17 **Velocity** decoding correlation coefficients comparison between the networks trained on non-whitened datasets **Graphs A, C, E, G** and the whitened dataset **Graphs B, D, F, H**. In all settings **A - E** the Deep4Net (k3_d3_sbp0) from Hammer et al., 2021 is labeled red.

Graphs A and **B** compare the performance of the networks when trained and validated on the full dataset. The stars in **B** denote CCs significantly lower compared to CCs of the same architecture in **A** (** p <0.01), (* p < 0.05), Wilcoxon signed rank test.

Graphs C and **D** show the correlation coefficients of the networks trained on full data and validated on low-passed data. The stars in **D** denote if the CCs are significantly lower compared to CCs of the same architecture in **C** (** p <0.01), (* p < 0.05), Wilcoxon signed rank test.

Graphs E and **F** compare the CCs of the CNNs when trained and validated on high-passed data. The stars in **F** denote CCs significantly lower compared to CCs of the same architecture in **E** (** p <0.01), (* p < 0.05), Wilcoxon signed rank test. **Graphs G** and **H** compare performance when trained on full data and validated on high-passed data. The stars in **H** denote CCs significantly lower compared to CCs of the same architecture in **G** (** p <0.01), (* p < 0.05), Wilcoxon signed rank test.

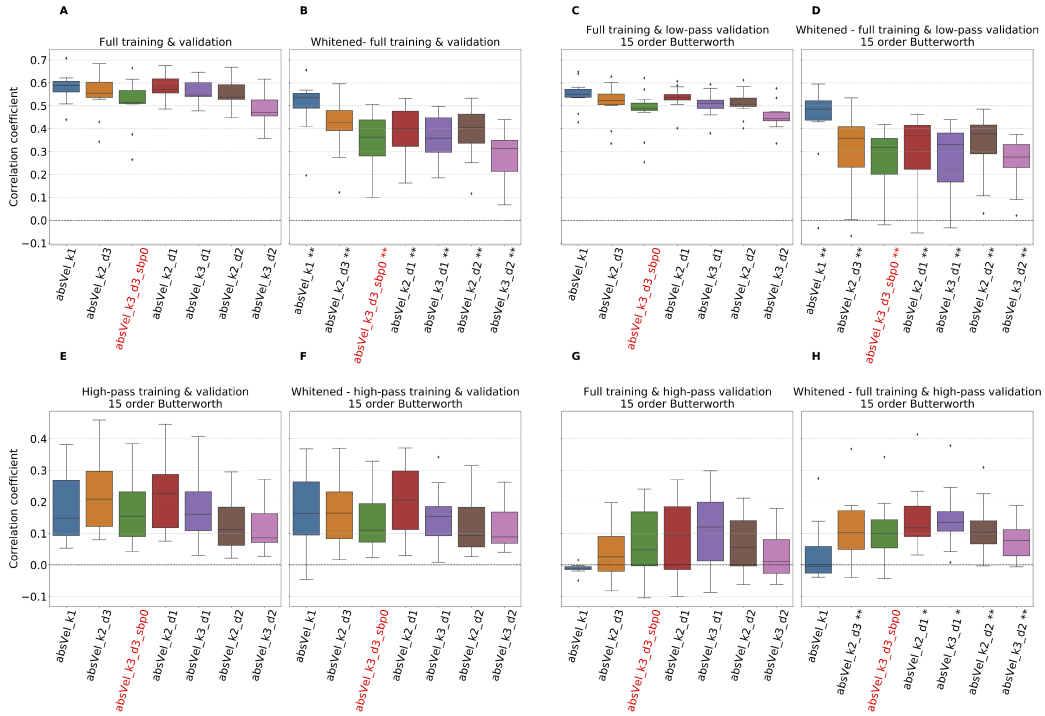


Figure 4.18 **Absolute velocity** decoding correlation coefficients comparison between the networks trained on the non-whitened datasets **Graphs A, C, E, G** and the whitened dataset **Graphs B, D, F, H**. In all settings **A - E** the Deep4Net (k3_d3_sbp0) from Hammer et al., 2021 is labeled red.

Graphs A and **B** compare the performance of the networks when trained and validated on the full dataset. The stars in **B** denote CCs significantly lower compared to CCs of the same architecture in **A**. (** p <0.01), (* p < 0.05), Wilcoxon signed rank test.

Graphs C and **D** show the correlation coefficients of the networks trained on full data and validated on low-passed data. The stars in **D** denote if the CCs are significantly lower compared to CCs of the same architecture in **C** (** p <0.01), (* p < 0.05), Wilcoxon signed rank test.

Graphs E and **F** compare the CCs of the CNNs when trained and validated on high-passed data. The stars in **F** denote CCs significantly lower compared to CCs of the same architecture in **E**. (** p <0.01), (* p < 0.05), Wilcoxon signed rank test. **Graphs G** and **H** compare performance when trained on full data and validated on high-passed data. The stars in **H** denote CCs significantly lower compared to CCs of the same architecture in **G**. (** p <0.01), (* p < 0.05), Wilcoxon signed rank test.

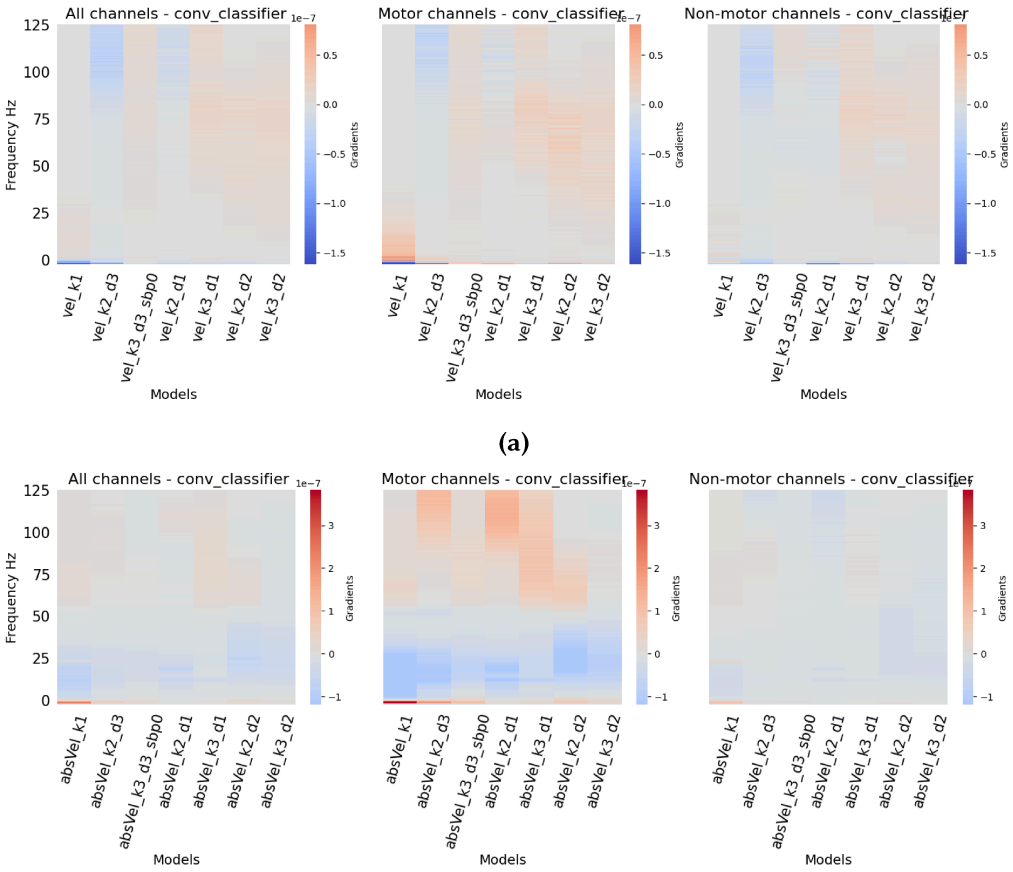


Figure 4.19 Gradients of networks trained to decode **(a)** velocity and **(b)** absolute velocity on spectrally-whitened datasets

from lower frequency bands in the signals. Furthermore, removing the max-pool layer helps the networks to utilise information in the low-frequency bands more effectively, even on the whitened dataset.

Conclusion

A deeper understanding of which iEEG features CNNs use for movement decoding broadens our understanding of how the brain encodes movement related information. It also partially alleviates the *black box* problem associated with deep neural networks. Improvement of performance of such networks makes them more perspective for real-life clinical applications. The work of Hammer et al., 2021 offers both performance improvement and a deeper understanding of which features are crucial for the CNN to make predictions. Nevertheless, their findings also raise some questions. Mainly the low importance of the high-gamma frequency band for absolute velocity decoding challenges previous findings about its informativeness for absolute velocity decoding (Hammer et al., 2016). The main goal of the thesis was to further study the frequency bands utilized by the Deep4Net with particular focus on the high-gamma frequencies and to identify modifications to the CNN architecture which improves utilization of information across useful frequency bands.

The contributions of this thesis are following:

- We showed that the high-gamma frequency band contains information relevant for movement decoding, especially for absolute velocity decoding. Nevertheless, its usage does not improve performance. On the contrary, its usage correlates with worse prediction when having access to all frequencies.

- We have identified the pooling layers in the architecture of the Deep4Net as unnecessary. Their removal optimizes information extraction from low frequency bands and improves overall performance of the CNN.

- We have identified the non-uniformity of the receptive field as a potential drawback of the architecture. It causes the CNN to be biased towards signals relatively far from the predicted time-point.

Main findings

Informative frequency bands

Most of the experiments we performed were motivated by making the network use information from the high-gamma frequency band in their prediction. Building different architectures was motivated by studying a peak in the high-gamma band, shifting the predicted time-point was motivated by the hypothesis, that relevant information is contained in the high-gamma of iEEG signals only directly before movement execution, and spectral whitening was motivated by a difference reduction in the amplitudes of lower vs. higher frequencies because we thought that lower amplitudes of high-frequencies are a disadvantage. Based on previous literature, the usage of high-gamma especially for absolute velocity decoding seemed like something that should occur. And because it did not, the idea was that forcing the network to use high-gamma would improve their performance.

Nevertheless, despite the substantial effort to show that high-gamma modulations are informative for velocity and absolute velocity decoding, the results repeatedly show that the Deep4Net and similar CNN architectures do not benefit from using high-gamma information when having access to information from all frequency bands. While we show that the high-gamma frequencies in iEEG contain relevant information especially for absolute velocity decoding, the CNNs profit from not using this information when making predictions. The most successful architecture, namely the CNN without max-pool layers, was the one least focused on modulations in the high-gamma frequency band. It was also the network which despite spectral whitening stayed focused mainly on low-frequency modulations and its performance dropped least. The shift of the predicted time-point to the centre of the receptive field which greatly improved performance also did not cause an increased interest in the high-gamma frequencies of any network. Rather it refined the focus of the networks to more narrow and mostly also lower frequency bands.

Architecture modifications

We have identified two main possibilities for architectural improvement of the Deep4Net. First, removing the max-pool layers from the architecture improves performance significantly compared to the original Deep4Net. Based on the performance evaluation on filtered datasets and the gradient visualization, we find that this modification allows the network to use information from low frequency bands more effectively.

Secondly, the non-uniformity of the receptive field could be a drawback of

the architecture. It causes the network to consider mostly information which are too far in the past with respect to the predicted time-point. We hypothesise, that making the receptive field uniform, by allowing padding at least in the direction from the centre of the receptive field towards the predicted time-point, would improve the achieved correlation coefficients. While the experiments we conducted in this thesis cannot clearly confirm this hypothesis, the performance improvement when shifting the predicted time-point to the centre of the receptive field speaks at least partially in its favor.

These two findings are useful for designing CNNs for movement decoding in the future.

Future work

Our research offers a wide range of future directions. The most prominent next step would be building a network with a uniform receptive field which would still be useful for on-line BCI and would likely make better predictions. Further steps could also be taken in studying which frequencies the network uses. Even though our research was fairly thorough in this regard, there is always more to find. The signal consist not only from amplitudes of different frequencies but also contains phase information which has been shown to also hold movement related information (Hammer et al., 2021; Hartmann et al., 2018). Thus, a similar analysis of the influence of phase of the different frequencies might be an interesting future project. And even when considering amplitudes again, besides the high-passed and low-passed datasets, datasets band-passed to only certain, more narrow, frequency bands could be created and the performance and gradients of the networks could be analysed similarly to what we did in our thesis.

Bibliography

Ahmadlou, M. et al. (2011). "Fractality and a wavelet-chaos-methodology for EEG-based diagnosis of Alzheimer disease". In: *Alzheimer Disease & Associated Disorders* 25.1, pp. 85–92.

Anderson, N. et al. (Mar. 2009). "An Offline Evaluation of the Autoregressive Spectrum for Electrocorticography". In: *IEEE Transactions on Biomedical Engineering* 56.3, pp. 913–916. ISSN: 0018-9294. DOI: 10.1109/TBME.2009.2009767. URL: http://ieeexplore.ieee.org/document/4838922/ (visited on 04/29/2021).

Angrick, M. et al. (Nov. 2018). "Speech Synthesis from ECoG using Densely Connected 3D Convolutional Neural Networks". en. In: *bioRxiv*, p. 478644. DOI: 10.1101/478644. URL: https://www.biorxiv.org/content/10.1101/478644v1 (visited on 11/16/2020).

Ball, T. et al. (July 2008). "Movement related activity in the high gamma range of the human EEG". In: *NeuroImage* 41, pp. 302–10. DOI: 10.1016/j.neuroimage.2008.02.032.

Ball, T. et al. (Mar. 2009). "Differential representation of arm movement direction in relation to cortical anatomy and function". In: *Journal of neural engineering* 6, p. 016006. DOI: 10.1088/1741-2560/6/1/016006.

Bottou, L. (2010). "Large-scale machine learning with stochastic gradient descent". In: *Proceedings of COMPSTAT'2010*. Springer, pp. 177–186.

Brock, A. et al. (2021). *High-Performance Large-Scale Image Recognition Without Normalization*. arXiv: 2102.06171 [cs.CV].

Brunel, N. et al. (Jan. 2008). "Lapicque's 1907 paper: From frogs to integrate-and-fire". In: *Biological cybernetics* 97, pp. 337–9. DOI: 10.1007/s00422-007-0190-0.

Butterworth, S. et al. (1930). "On the theory of filter amplifiers". In: *Wireless Engineer* 7.6, pp. 536–541.

Buzsaki, G. et al. (June 2012). "The origin of extracellular fields and currents — EEG, ECoG, LFP and spikes". en. In: *Nature Reviews Neuroscience* 13.6, pp. 407–420. ISSN: 1471-003X, 1471-0048. DOI: 10.1038/nrn3241. URL: http://www.nature.com/articles/nrn3241 (visited on 04/07/2021).

Clevert, D.-A. et al. (Feb. 2016). "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)". In: *arXiv:1511.07289 [cs]*. arXiv: 1511.07289. URL: http://arxiv.org/abs/1511.07289 (visited on 04/24/2021).

Delgado Saa, J. (May 2020). "Using Coherence-based spectro-spatial filters for stimulus features prediction from electro-corticographic recordings". In: *Scientific Reports*. DOI: 10.1038/s41598-020-63303-1.

Duchi, J. et al. (2011). "Adaptive subgradient methods for online learning and stochastic optimization." In: *Journal of machine learning research* 12.7.

Dumoulin, V. et al. (Jan. 2018). "A guide to convolution arithmetic for deep learning". In: *arXiv:1603.07285 [cs, stat]*. arXiv: 1603.07285. URL: http://arxiv.org/abs/1603.07285 (visited on 04/28/2021).

Edelman, B. J. et al. (2016). "EEG Source Imaging Enhances the Decoding of Complex Right-Hand Motor Imagery Tasks". In: *IEEE Transactions on Biomedical Engineering* 63.1, pp. 4–14. DOI: 10.1109/TBME.2015.2467312.

Eickhoff, S. B. et al. (May 2005). "A new SPM toolbox for combining probabilistic cytoarchitectonic maps and functional imaging data". en. In: *NeuroImage* 25.4, pp. 1325–1335. ISSN: 10538119. DOI: 10.1016/j.neuroimage.2004.12.034. URL: https://linkinghub.elsevier.com/retrieve/pii/S105381190400792X (visited on 04/24/2021).

Erhan, D. et al. (Jan. 2009). "Visualizing Higher-Layer Features of a Deep Network". In: *Technical Report, Univeristé de Montréal*.

Friedman, J. H. (1989). "Regularized discriminant analysis". In: *Journal of the American statistical association* 84.405, pp. 165–175.

Goodfellow, S. et al. (2018). "Towards Understanding ECG Rhythm Classification Using Convolutional Neural Networks and Attention Mappings". In: *MLHC*.

Gu, J. et al. (2018). "Recent advances in convolutional neural networks". In: *Pattern Recognition* 77, pp. 354–377. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2017.10.013. URL: https://www.sciencedirect.com/science/article/pii/S0031320317304120.

Gunduz, A. et al. (2016). "Differential roles of high gamma and local motor potentials for movement preparation and execution". In: *Brain-Computer Interfaces* 3.2, pp. 88–102. DOI: 10.1080/2326263X.2016.1179087. eprint: https://doi.org/10.1080/2326263X.2016.1179087. URL: https://doi.org/10.1080/2326263X.2016.1179087.

Hammer, J. et al. (2013). "The role of ECoG magnitude and phase in decoding position, velocity, and acceleration during continuous motor behavior". English. In: *Frontiers in Neuroscience* 7. ISSN: 1662-453X. DOI: 10.3389/fnins.2013.00200. URL: https://www.frontiersin.org/articles/10.3389/fnins.2013.00200/full (visited on 04/24/2021).

Hammer, J. et al. (June 2016). "Predominance of Movement Speed Over Direction in Neuronal Population Signals of Motor Cortex: Intracranial EEG Data and

A Simple Explanatory Model". en. In: *Cerebral Cortex* 26.6, pp. 2863–2881. ISSN: 1047-3211, 1460-2199. DOI: 10.1093/cercor/bhw033. URL: https://academic.oup.com/cercor/article-lookup/doi/10.1093/cercor/bhw033 (visited on 01/21/2021).

Hammer, J. et al. (2021). "Functional Integration and Segregation Emerging in Deep Convolutional Networks Trained for Brain Signal Decoding". unpublished.

Hartmann, K. G. et al. (Jan. 2018). "Hierarchical internal representation of spectral features in deep convolutional networks trained for EEG decoding". In: *2018 6th International Conference on Brain-Computer Interface (BCI)*. arXiv: 1711.07792, pp. 1–6. DOI: 10.1109/IWW-BCI.2018.8311493. URL: http://arxiv.org/abs/1711.07792 (visited on 11/16/2020).

Hassan, A. R. et al. (2016). "Epileptic seizure detection in EEG signals using tunable-Q factor wavelet transform and bootstrap aggregating". In: *Computer Methods and Programs in Biomedicine* 137, pp. 247–259. ISSN: 0169-2607. DOI: https://doi.org/10.1016/j.cmpb.2016.09.008. URL: https://www.sciencedirect.com/science/article/pii/S0169260716304370.

Hearst, M. et al. (1998). "Support vector machines". In: *IEEE Intelligent Systems and their Applications* 13.4, pp. 18–28. DOI: 10.1109/5254.708428.

Hinton, G. E. et al. (2012). "Improving neural networks by preventing co-adaptation of feature detectors". In: *arXiv preprint arXiv:1207.0580*.

Hochreiter, S. et al. (Nov. 1997). "Long Short-Term Memory". In: *Neural Comput.* 9.8, 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. URL: https://doi.org/10.1162/neco.1997.9.8.1735.

Hodgkin, A. L. et al. (1952). "A quantitative description of membrane current and its application to conduction and excitation in nerve". In: *The Journal of physiology* 117.4, pp. 500–544.

Hotson, G. et al. (2014). "Coarse Electrocorticographic Decoding of Ipsilateral Reach in Patients with Brain Lesions". In: *PLoS ONE* 9.

Hotson, G. et al. (2016). "Individual finger control of a modular prosthetic limb using high-density electrocorticography in a human subject". In: *Journal of neural engineering* 13.2, p. 026017.

Ioffe, S. et al. (2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *International conference on machine learning*. PMLR, pp. 448–456.

Kalchbrenner, N. et al. (2016). "Neural Machine Translation in Linear Time". In: *CoRR* abs/1610.10099. arXiv: 1610.10099. URL: http://arxiv.org/abs/1610.10099.

Kingma, D. P. et al. (Jan. 2017). "Adam: A Method for Stochastic Optimization". In: *arXiv:1412.6980 [cs]*. arXiv: 1412.6980. URL: http://arxiv.org/abs/1412.6980 (visited on 04/25/2021).

Krizhevsky, A. et al. (2012). "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems* 25, pp. 1097–1105.

Lawhern, V. J. et al. (Oct. 2018). "EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces". In: *Journal of Neural Engineering* 15.5, p. 056013. ISSN: 1741-2560, 1741-2552. DOI: 10.1088/1741-2552/aace8c. URL: https://iopscience.iop.org/article/10.1088/1741-2552/aace8c (visited on 11/16/2020).

Lebedev, M. A. et al. (May 2005). "Cortical Ensemble Adaptation to Represent Velocity of an Artificial Actuator Controlled by a Brain-Machine Interface". en. In: *Journal of Neuroscience* 25.19, pp. 4681–4693. ISSN: 0270-6474, 1529-2401. DOI: 10.1523/JNEUROSCI.4088-04.2005. URL: https://www.jneurosci.org/content/25/19/4681 (visited on 04/08/2021).

Li, G. et al. (2018). "Optimal referencing for stereo-electroencephalographic (SEEG) recordings". In: *NeuroImage* 183, pp. 327–335. ISSN: 1053-8119. DOI: https://doi.org/10.1016/j.neuroimage.2018.08.020.

Light, G. et al. (July 2010). "Electroencephalography (EEG) and Event-Related Potentials (ERPs) with Human Participants". In: *Current protocols in neuroscience / editorial board, Jacqueline N. Crawley ... [et al.]* Chapter 6, Unit 6.25.1–24. DOI: 10.1002/0471142301.ns0625s52.

Lipton, Z. C. (2015). "A Critical Review of Recurrent Neural Networks for Sequence Learning". In: *CoRR* abs/1506.00019. arXiv: 1506.00019. URL: http://arxiv.org/abs/1506.00019.

Liu, Y et al. (Oct. 2015). "The effects of spatial filtering and artifacts on electrocorticographic signals". In: *Journal of Neural Engineering* 12.5, p. 056008. ISSN: 1741-2560, 1741-2552. DOI: 10.1088/1741-2560/12/5/056008. URL: https://iopscience.iop.org/article/10.1088/1741-2560/12/5/056008 (visited on 04/08/2021).

Long, J. et al. (2015). "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440.

Lotte, F. et al. (2018). "A review of classification algorithms for EEG-based brain–computer interfaces: a 10 year update". In: *Journal of neural engineering* 15.3, p. 031005.

Lu, J. et al. (2012). "Adaptive Laplacian filtering for sensorimotor rhythm-based brain-computer interfaces". In: *Journal of neural engineering* 10, p. 016002. DOI: 10.1088/1741-2560/10/1/016002.

Luu, T. P. et al. (Aug. 2016). "Unscented Kalman Filter for Neural Decoding of Human Treadmill Walking from Non-invasive Electroencephalography". In: vol. 2016. DOI: 10.1109/EMBC.2016.7591006.

Lv, J. et al. (2010). “Decoding hand movement velocity from electroencephalogram signals during a drawing task”. In: *Biomedical engineering online* 9.1, pp. 1–21.

Michelmann, S. et al. (2018). “Data-driven re-referencing of intracranial EEG based on independent component analysis (ICA)”. In: *Journal of Neuroscience Methods* 307, pp. 125–137. ISSN: 0165-0270. DOI: https://doi.org/10.1016/j.jneumeth.2018.06.021. URL: https://www.sciencedirect.com/science/article/pii/S0165027018301997.

Mika, S. et al. (1999). “Fisher discriminant analysis with kernels”. In: *Neural Networks for Signal Processing IX: Proceedings of the 1999 IEEE Signal Processing Society Workshop (Cat. No.98TH8468)*, pp. 41–48. DOI: 10.1109/NNSP.1999.788121.

Milstein, D. et al. (2017). “Multiscale semi-markov dynamics for intracortical brain-computer interfaces”. In: *Advances in Neural Information Processing Systems* 30, pp. 868–878.

Mousavi, S. et al. (Mar. 2019). “SleepEEGNet: Automated Sleep Stage Scoring with Sequence to Sequence Deep Learning Approach”. In.

Nair, V. et al. (2010). “Rectified linear units improve restricted boltzmann machines”. In: *Icml*.

Ofner, P. et al. (2012). “Decoding of velocities and positions of 3D arm movement from EEG”. In: *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 6406–6409. DOI: 10.1109/EMBC.2012.6347460.

OShea, K. et al. (2015). “An Introduction to Convolutional Neural Networks”. In: *CoRR* abs/1511.08458. arXiv: 1511.08458. URL: http://arxiv.org/abs/1511.08458.

Pearson, K. (1901). “LIII. On lines and planes of closest fit to systems of points in space”. In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11, pp. 559–572.

Pearson, K. et al. (Jan. 1895). “VII. Note on regression and inheritance in the case of two parents”. In: *Proceedings of the Royal Society of London* 58.347-352, pp. 240–242. DOI: 10.1098/rspl.1895.0041. URL: https://royalsocietypublishing.org/doi/10.1098/rspl.1895.0041 (visited on 04/27/2021).

Pistohl, T. et al. (2008). “Prediction of arm movement trajectories from ECoG-recordings in humans”. In: *Journal of Neuroscience Methods* 167, pp. 105–114.

Pistohl, T. et al. (July 2011). “Decoding natural grasp types from human ECoG”. In: *NeuroImage* 59, pp. 248–60. DOI: 10.1016/j.neuroimage.2011.06.084.

Rae, J. W. et al. (2019). *Compressive Transformers for Long-Range Sequence Modelling*. arXiv: 1911.05507 [cs.LG].

Rieke, J. et al. (2018). "Visualizing Convolutional Networks for MRI-Based Diagnosis of Alzheimer's Disease". en. In: *Understanding and Interpreting Machine Learning in Medical Image Computing Applications*. Ed. by D. Stoyanov et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 24–31. ISBN: 9783030026288. DOI: 10.1007/978-3-030-02628-8_3.

Roy, Y. et al. (Aug. 2019). "Deep learning-based electroencephalography analysis: a systematic review". In: *Journal of Neural Engineering* 16.5, p. 051001. DOI: 10.1088/1741-2552/ab260c. URL: https://doi.org/10.1088/1741-2552/ab260c.

Rumelhart, D. E. et al. (1987). "Learning Internal Representations by Error Propagation". In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*, pp. 318–362.

Saa, J. D. et al. (2016). "Asynchronous decoding of finger movements from ECoG signals using long-range dependencies conditional random fields". In.

Satow, T. et al. (2003). "Distinct cortical areas for motor preparation and execution in human identified by Bereitschaftspotential recording and ECoG-EMG coherence analysis". In: *Clinical neurophysiology* 114.7, pp. 1259–1264.

Schalk, G. et al. (Oct. 2007). "Decoding two-dimensional movement trajectories using electrocorticographic signals in humans". In: *Journal of neural engineering* 4, pp. 264–75. DOI: 10.1088/1741-2560/4/3/012.

Schalk, G. et al. (2011). "Brain-Computer Interfaces Using Electrocorticographic Signals". In: *IEEE Reviews in Biomedical Engineering* 4, pp. 140–154. DOI: 10.1109/RBME.2011.2172408.

Schirrmeister, R. T. et al. (Nov. 2017). "Deep learning with convolutional neural networks for EEG decoding and visualization: Convolutional Neural Networks in EEG Analysis". en. In: *Human Brain Mapping* 38.11, pp. 5391–5420. ISSN: 10659471. DOI: 10.1002/hbm.23730. URL: http://doi.wiley.com/10.1002/hbm.23730 (visited on 11/16/2020).

Scholg, A. et al. (2002). *Presence Research and EEG*. en. URL: https://www.semanticscholar.org/paper/Presence-Research-and-EEG-Scholg-Slater/18816255d88653d9bbaedb7a24c4394a6663c7a7.

Schomer, D. L. et al. (Nov. 2017). *Niedermeyer's ElectroencephalographyBasic Principles, Clinical Applications, and Related Fields: Basic Principles, Clinical Applications, and Related Fields*. Oxford, UK: Oxford University Press. ISBN: 9780190228514. DOI: 10.1093/med/9780190228484.001.0001. URL: https://oxfordmedicine.com/view/10.1093/med/9780190228484.001.0001/med-9780190228484.

Şen, B. et al. (2014). "A comparative study on classification of sleep stage based on EEG signals using feature selection and classification algorithms". In: *Journal of medical systems* 38.3, pp. 1–21.

Simonyan, K. et al. (2014). “Deep inside convolutional networks: Visualising image classification models and saliency maps”. In.

Sturm, I. et al. (Apr. 2016). “Interpretable Deep Neural Networks for Single-Trial EEG Classification”. In: *arXiv:1604.08201 [cs, stat]*. arXiv: 1604.08201. URL: http://arxiv.org/abs/1604.08201 (visited on 11/16/2020).

Tam, W.-k. et al. (Sept. 2019). “Human motor decoding from neural signals: a review”. In: *BMC Biomedical Engineering* 1.1, p. 22. ISSN: 2524-4426. DOI: 10.1186/s42490-019-0022-z. URL: https://doi.org/10.1186/s42490-019-0022-z (visited on 04/07/2021).

Tsai, Y.-C. et al. (2020). “Predict Forex Trend via Convolutional Neural Networks”. In: *Journal of Intelligent Systems* 29.1, pp. 941–958. DOI: doi:10.1515/jisys-2018-0074. URL: https://doi.org/10.1515/jisys-2018-0074.

Van Vugt, M. et al. (Jan. 2007). “Comparison of spectral analysis methods for characterizing brain oscillations”. In: *Journal of Neuroscience Methods* 162, pp. 49–63.

Volkova, K. et al. (Dec. 2019). “Decoding Movement From Electrocorticographic Activity: A Review”. In: *Frontiers in Neuroinformatics* 13, p. 74. DOI: 10.3389/fninf.2019.00074.

Walter, A. et al. (2012). “Coupling BCI and cortical stimulation for brain-state-dependent stimulation: methods for spectral estimation in the presence of stimulation after-effects”. In: *Frontiers in Neural Circuits* 6, p. 87. ISSN: 1662-5110. DOI: 10.3389/fncir.2012.00087. URL: https://www.frontiersin.org/article/10.3389/fncir.2012.00087.

Wang, Z et al. (Nov. 2011). “Prior Knowledge Improves Decoding of Finger Flexion from Electrocorticographic Signals”. In: *Frontiers in neuroscience* 5, p. 127. DOI: 10.3389/fnins.2011.00127.

Wen, T. et al. (2018). “Deep Convolution Neural Network and Autoencoders-Based Unsupervised Feature Learning of EEG Signals”. In: *IEEE Access* 6, pp. 25399–25410. DOI: 10.1109/ACCESS.2018.2833746.

Wolpaw, J. R. et al. (1991). “An EEG-based brain-computer interface for cursor control”. In: *Electroencephalography and Clinical Neurophysiology* 78.3, pp. 252–259. ISSN: 0013-4694. DOI: https://doi.org/10.1016/0013-4694(91)90040-B. URL: https://www.sciencedirect.com/science/article/pii/001346949190040B.

Xie, Z. et al. (Nov. 2017). “Decoding of finger trajectory from ECoG using Deep Learning”. In: *Journal of Neural Engineering* 15. DOI: 10.1088/1741-2552/aa9dbe.

Yang, C. et al. (July 2018). “Visual Explanations From Deep 3D Convolutional Neural Networks for Alzheimer’s Disease Classification”. In: *arXiv:1803.02544 [cs, stat]*. arXiv: 1803.02544. URL: http://arxiv.org/abs/1803.02544 (visited on 11/19/2020).

Yao, D. et al. (2019). "Which Reference Should We Use for EEG and ERP practice?" In: *Brain Topography* 32.4, p. 530.

Zhang, R. et al. (Oct. 2019). "A novel hybrid deep learning scheme for four-class motor imagery classification". In: *Journal of Neural Engineering* 16.6, p. 066004. DOI: 10.1088/1741-2552/ab3471. URL: https://doi.org/10.1088/1741-2552/ab3471.

Zheng, W.-L. et al. (2015). "Investigating critical frequency bands and channels for EEG-based emotion recognition with deep neural networks". In: *IEEE Transactions on Autonomous Mental Development* 7.3, pp. 162–175.

Zhou, B. et al. (2015). "Learning Deep Features for Discriminative Localization". In: *CoRR* abs/1512.04150. arXiv: 1512.04150. URL: http://arxiv.org/abs/1512.04150.

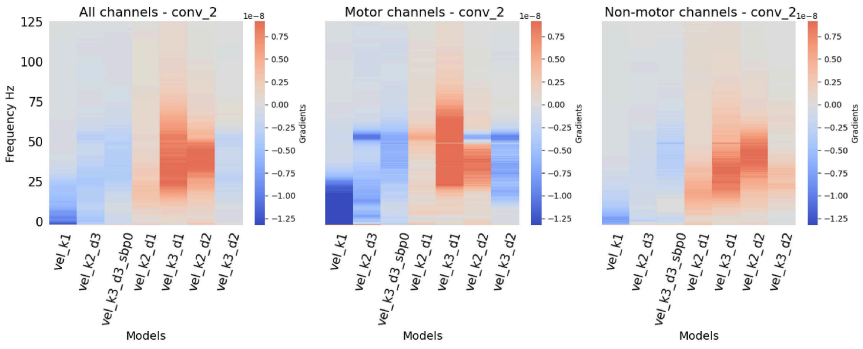
List of Figures

List of Tables

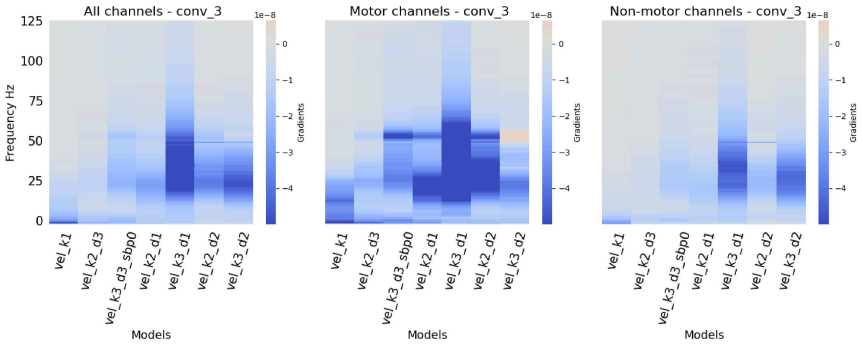
Appendix A

Gradients of all architectures - original setting (non-shifted)

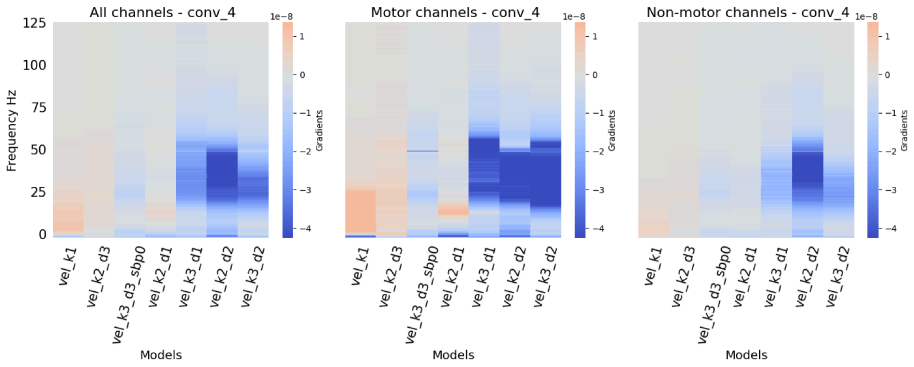
Velocity - Full dataset



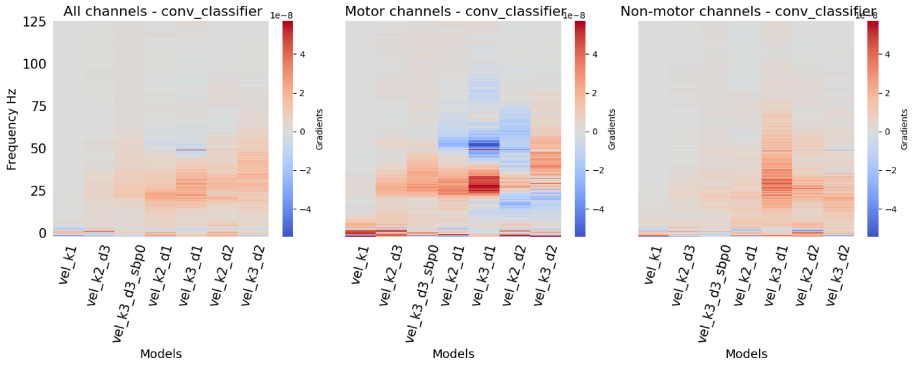
(a)



(b)



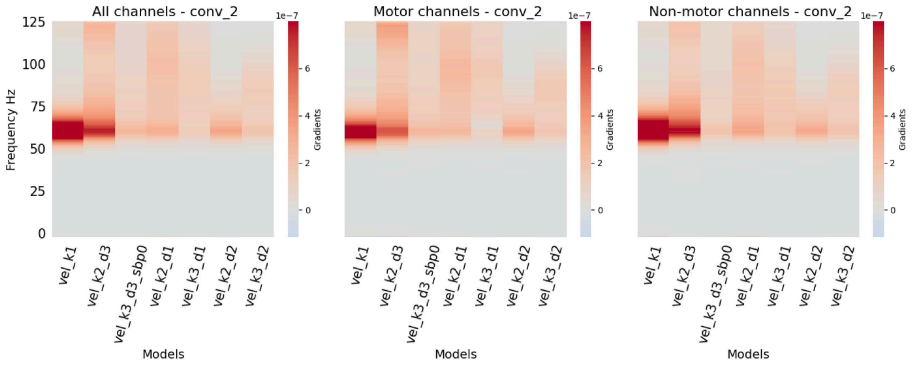
(c)



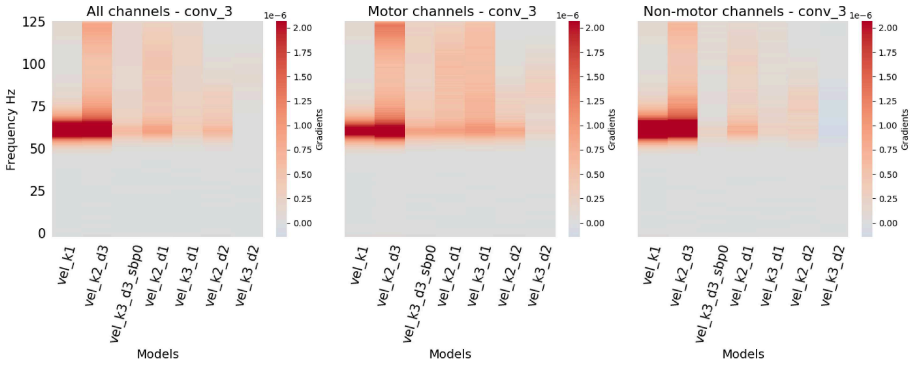
(d)

Figure A.1 Gradients of the different CNN architectures decoding velocity from the full dataset in the original non-shifted setting (causal prediction). **(a)** shows gradients of the convolutional layer in the second block; **(b)** shows gradients of the convolutional layer in the third block; **(c)** shows gradients of the fourth convolutional block; **(d)** shows gradients of the last convolutional layer - the output layer. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4

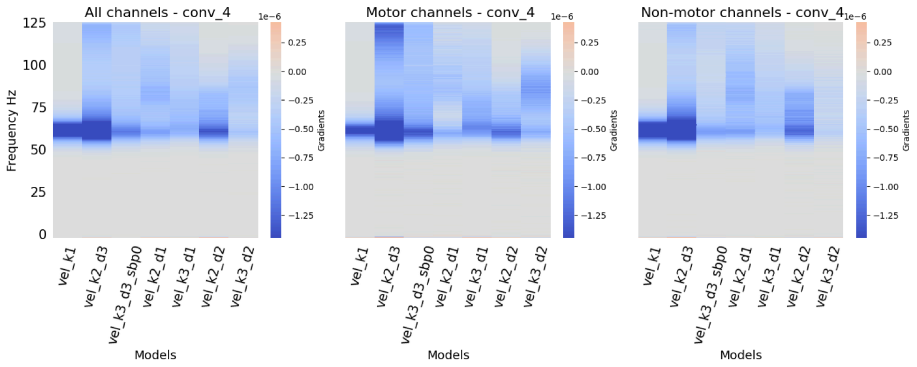
Velocity - High-passed dataset



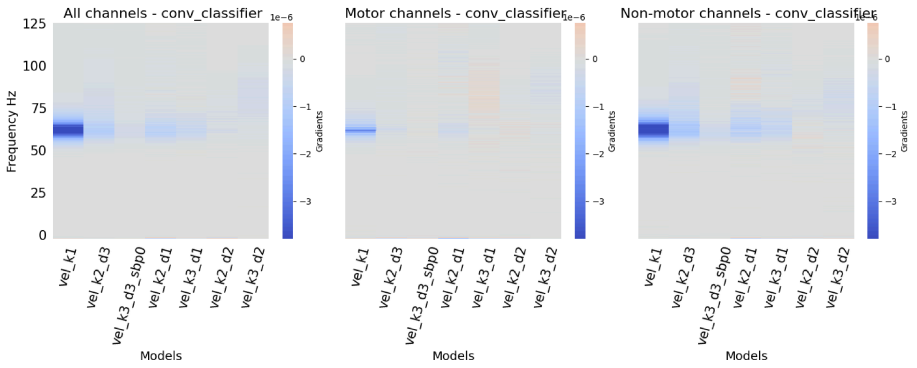
(a)



(b)



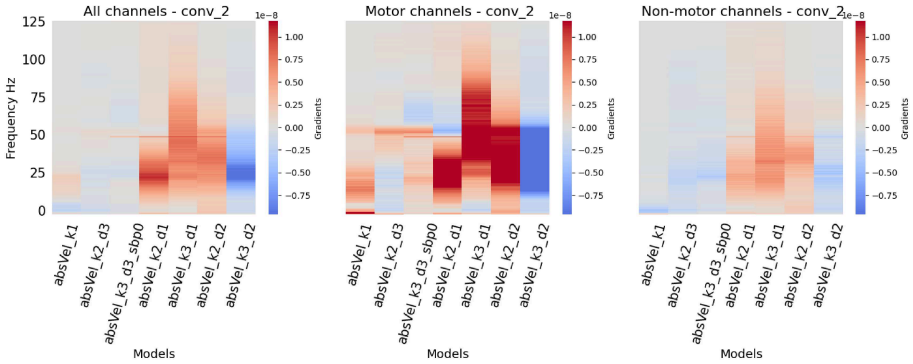
(c)



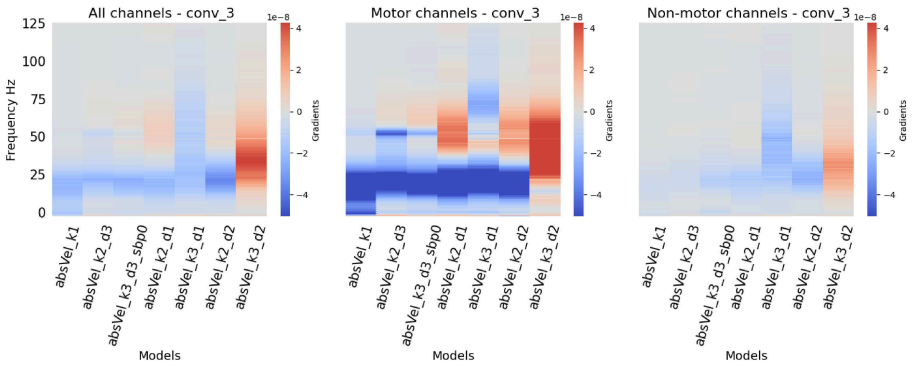
(d)

Figure A.2 Gradients of the different CNN architectures decoding velocity from the high-passed dataset in the original non-shifted setting (causal prediction). **(a)** shows gradients of the convolutional layer in the second block; **(b)** shows gradients of the convolutional layer in the third block; **(c)** shows gradients of the fourth convolutional block; **(d)** shows gradients of the last convolutional layer - the output layer. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4

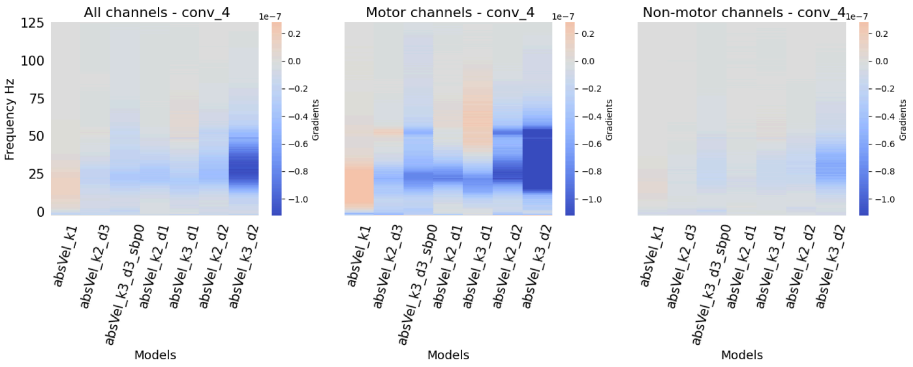
Absolute velocity - Full dataset



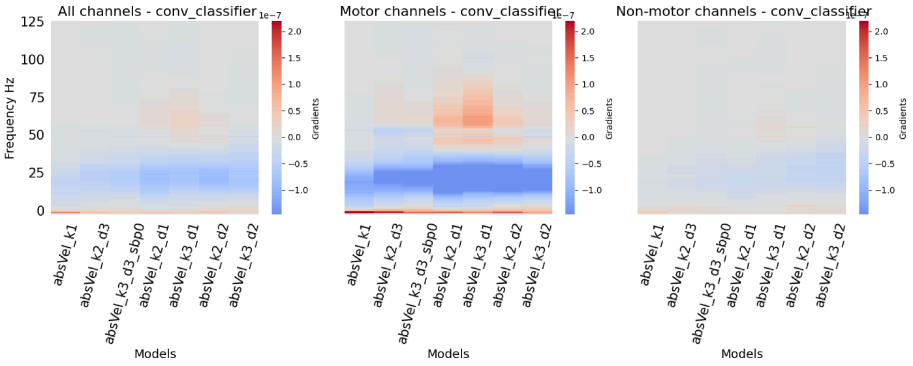
(a)



(b)



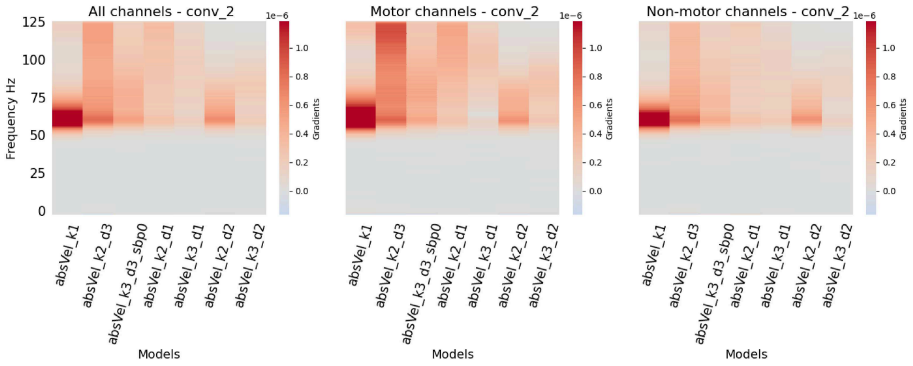
(c)



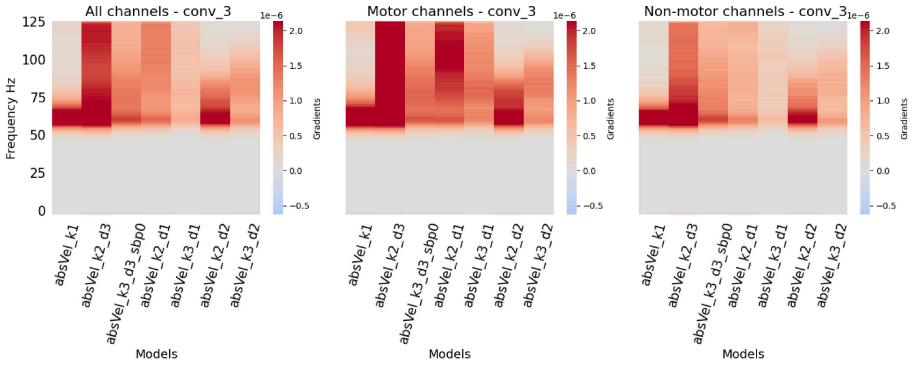
(d)

Figure A.3 Gradients of the different CNN architectures decoding absolute velocity from the full dataset in the original non-shifted setting (causal prediction). **(a)** shows gradients of the convolutional layer in the second block; **(b)** shows gradients of the convolutional layer in the third block; **(c)** shows gradients of the fourth convolutional block; **(d)** shows gradients of the last convolutional layer - the output layer. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4

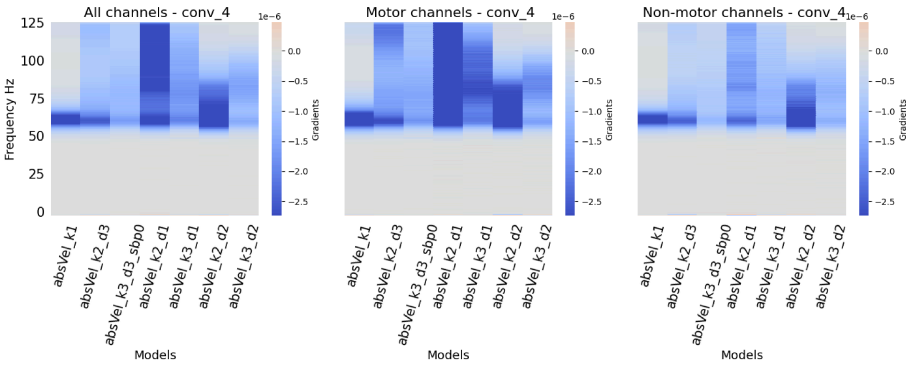
Absolute velocity - High-passed dataset



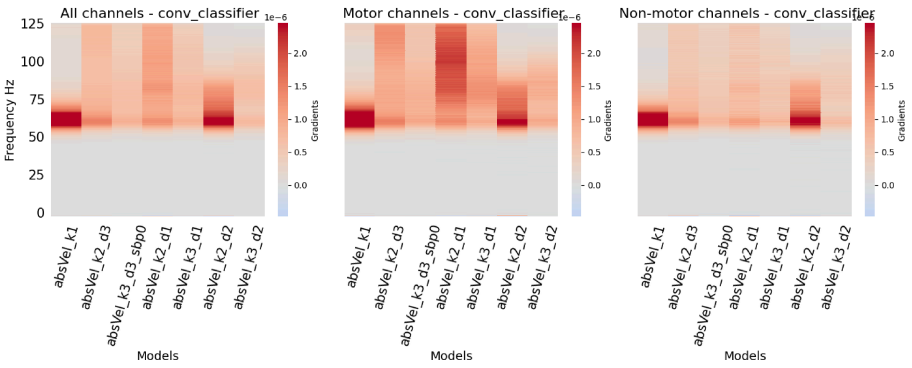
(a)



(b)



(c)



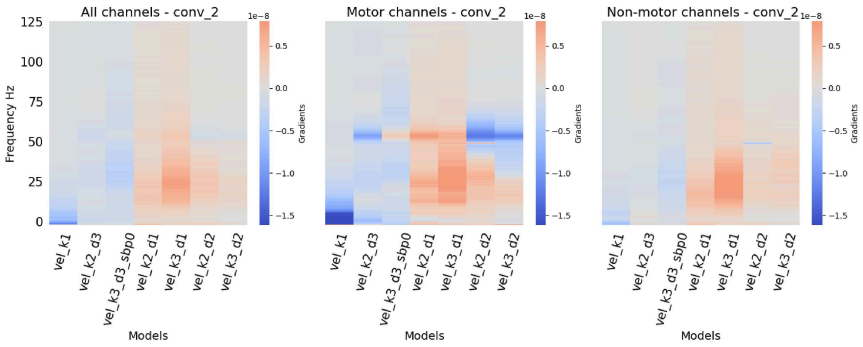
(d)

Figure A.4 Gradients of the different CNN architectures decoding absolute velocity from the high-passed dataset in the original non-shifted setting (causal prediction). **(a)** shows gradients of the convolutional layer in the second block; **(b)** shows gradients of the convolutional layer in the third block; **(c)** shows gradients of the fourth convolutional block; **(d)** shows gradients of the last convolutional layer - the output layer. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4

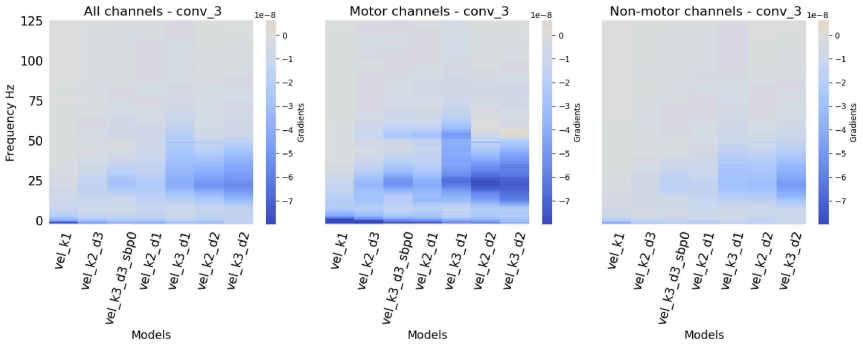
Appendix B

Gradients of all architectures - shifted

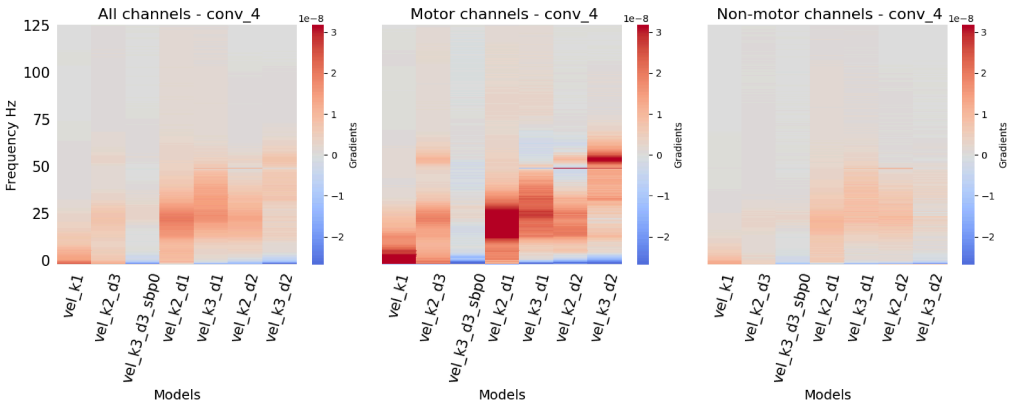
Velocity - Full dataset



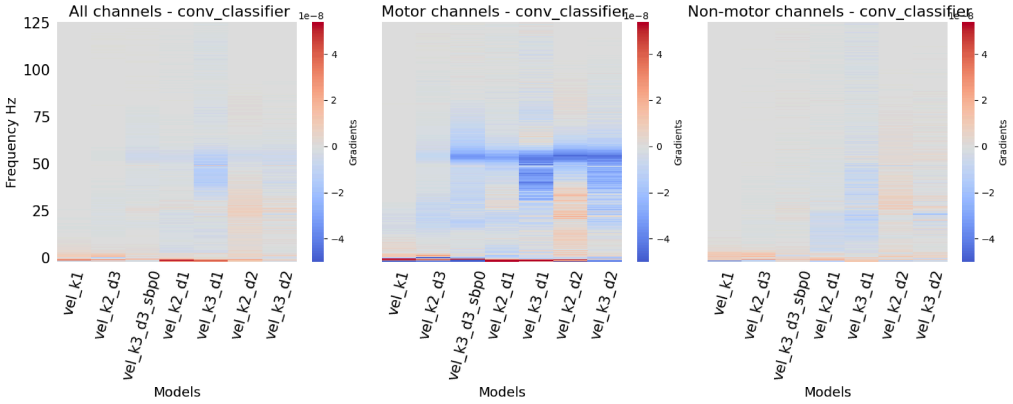
(a)



(b)



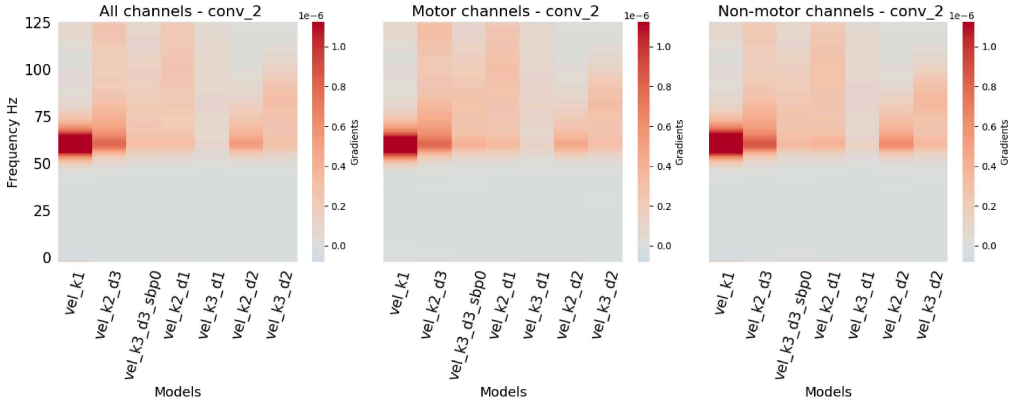
(c)



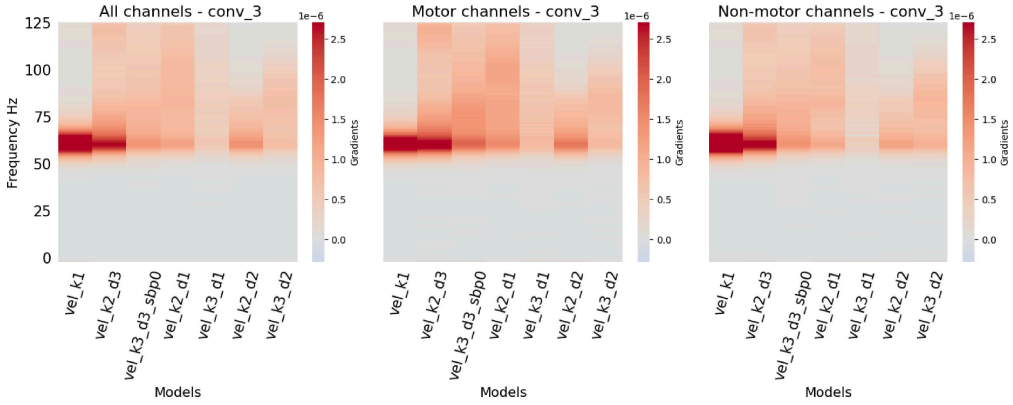
(d)

Figure B.1 Gradients of the different CNN architectures decoding velocity from the full dataset in the shifted setting (acausal prediction) (see Section 4.2.1. **(a)** shows gradients of the convolutional layer in the second block; **(b)** shows gradients of the convolutional layer in the third block; **(c)** shows gradients of the fourth convolutional block; **(d)** shows gradients of the last convolutional layer - the output layer. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4

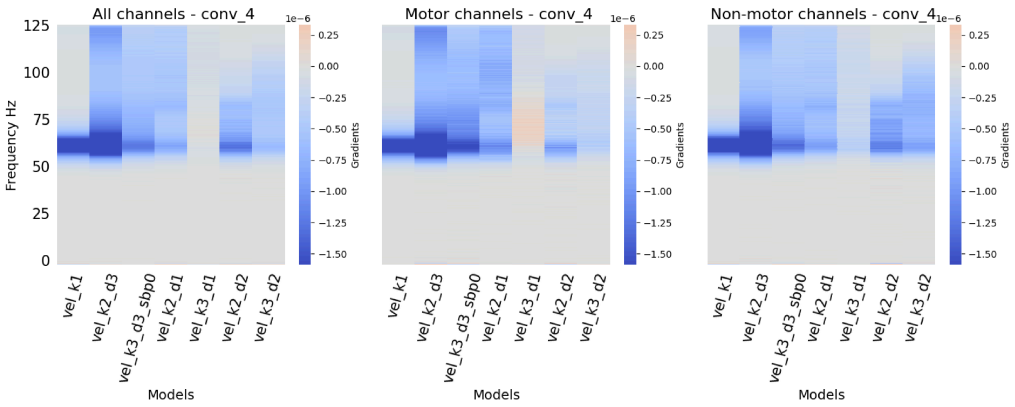
Velocity - High-passed dataset



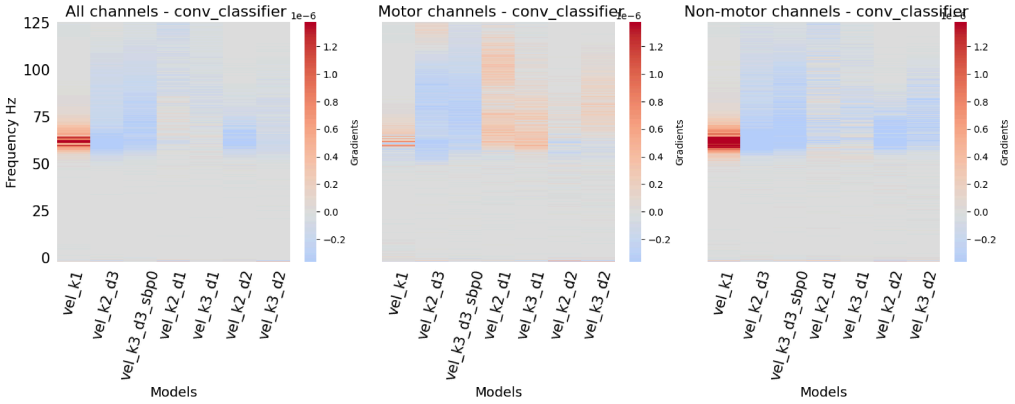
(a)



(b)



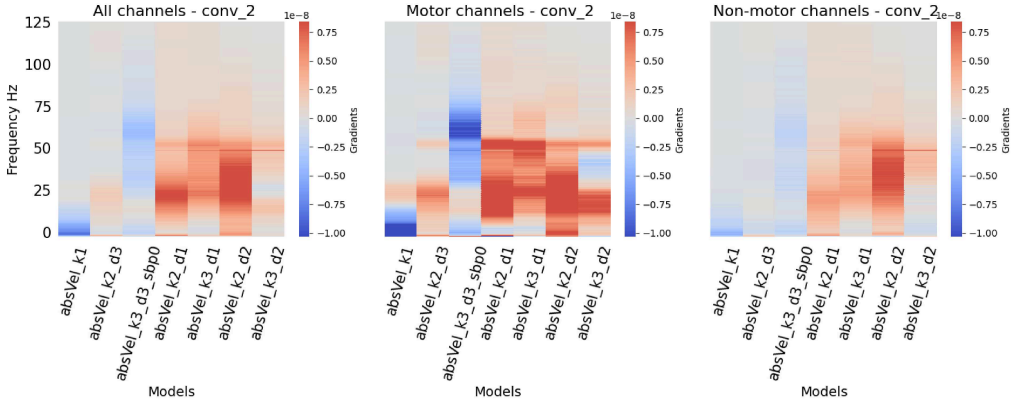
(c)



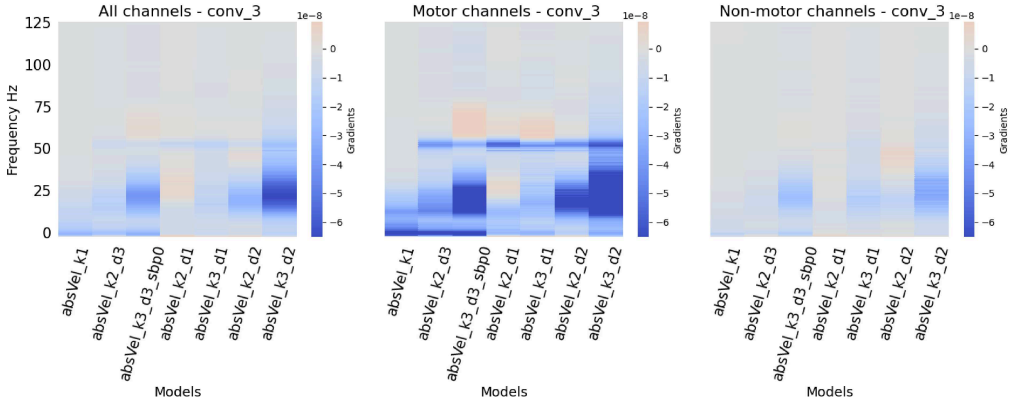
(d)

Figure B.2 Gradients of the different CNN architectures decoding velocity from the high-passed dataset in the shifted setting (acausal prediction) (see Section 4.2.1. **(a)** shows gradients of the convolutional layer in the second block; **(b)** shows gradients of the convolutional layer in the third block; **(c)** shows gradients of the fourth convolutional block; **(d)** shows gradients of the last convolutional layer - the output layer. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4

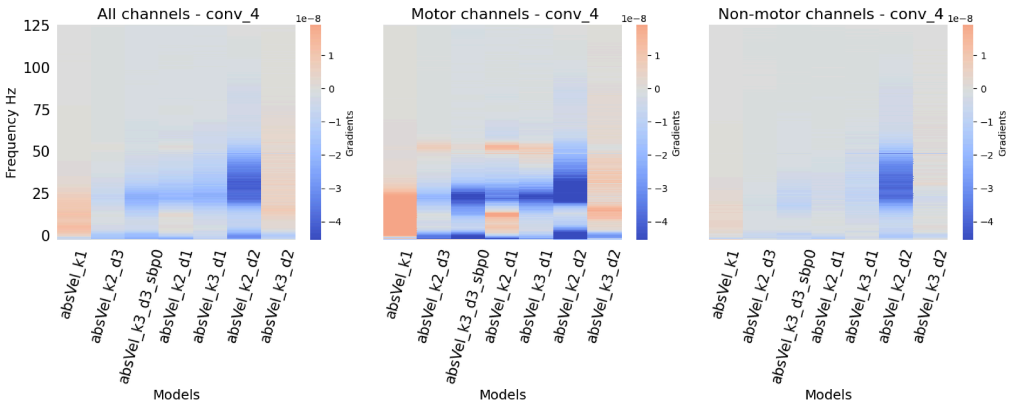
Absolute velocity - Full dataset



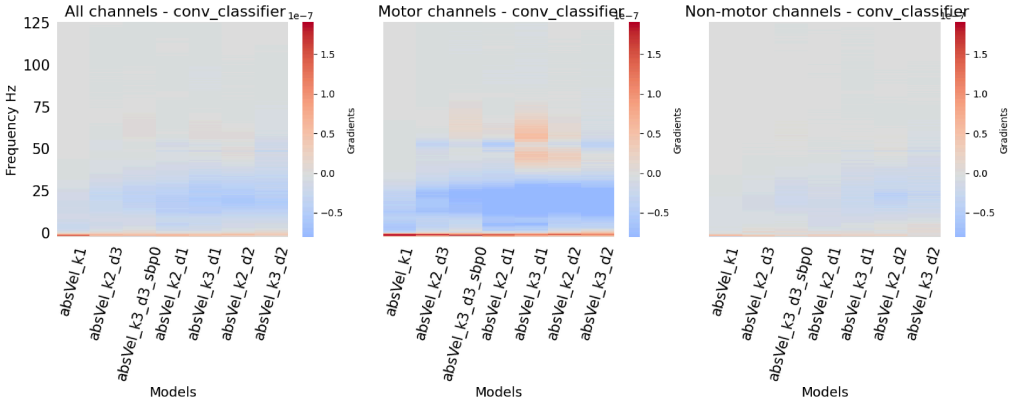
(a)



(b)



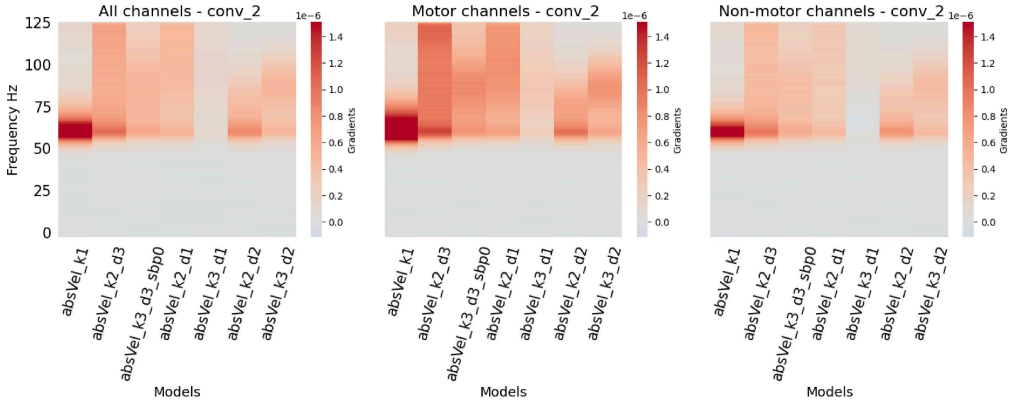
(c)



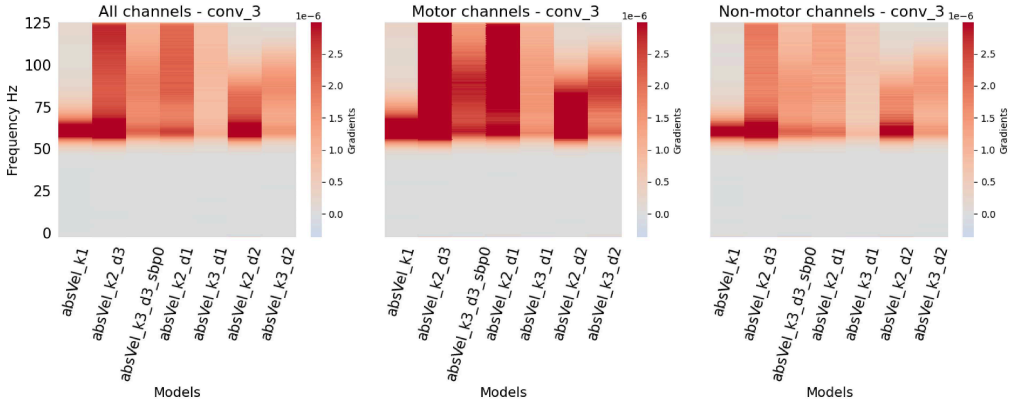
(d)

Figure B.3 Gradients of the different CNN architectures decoding absolute velocity from the full dataset in the shifted setting (acausal prediction) (see Section 4.2.1). **(a)** shows gradients of the convolutional layer in the second block; **(b)** shows gradients of the convolutional layer in the third block; **(c)** shows gradients of the fourth convolutional block; **(d)** shows gradients of the last convolutional layer - the output layer. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4

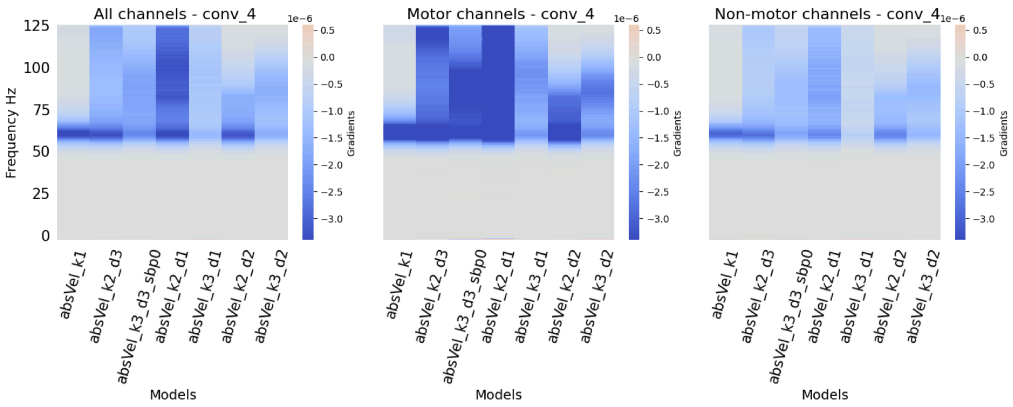
Absolute velocity - High-passed dataset



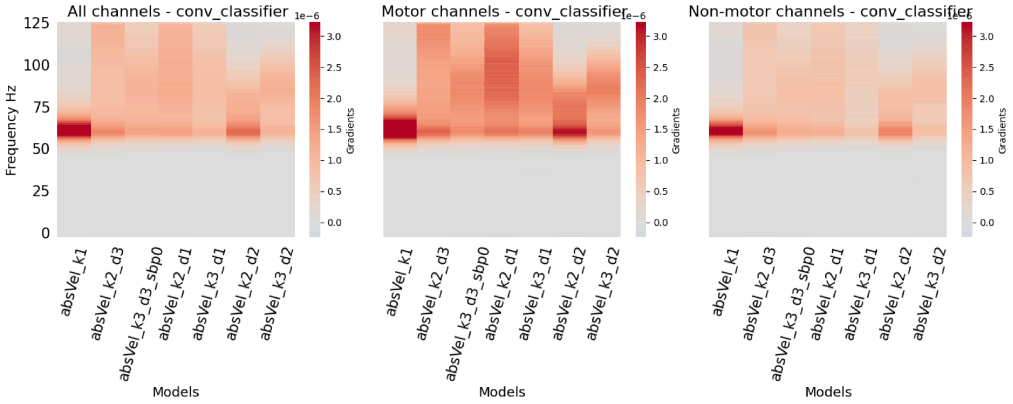
(a)



(b)



(c)



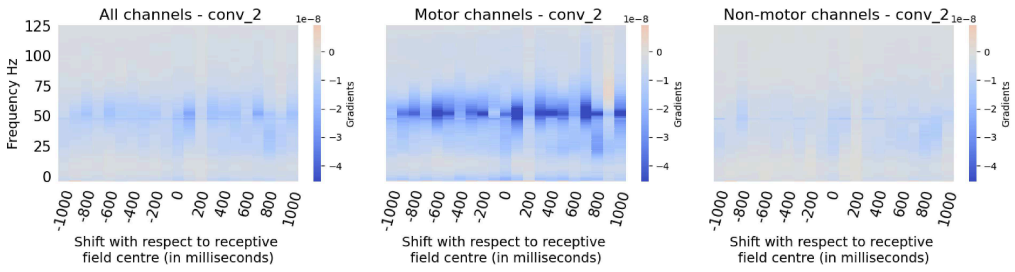
(d)

Figure B.4 Gradients of the different CNN architectures decoding absolute velocity from the high-passed dataset in the shifted setting (acausal prediction) (see Section 4.2.1). **(a)** shows gradients of the convolutional layer in the second block; **(b)** shows gradients of the convolutional layer in the third block; **(c)** shows gradients of the fourth convolutional block; **(d)** shows gradients of the last convolutional layer - the output layer. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4

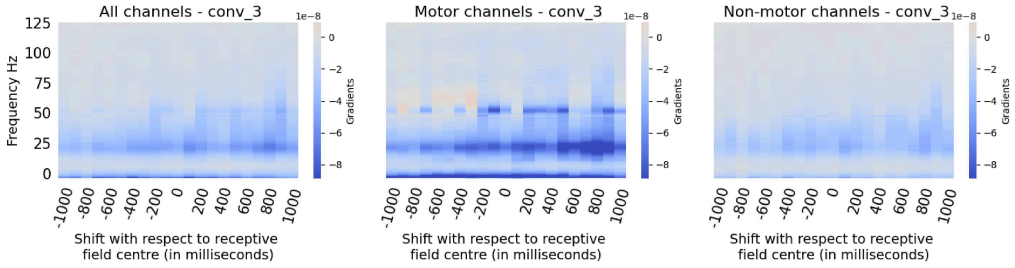
Appendix C

Gradients of all architectures - gradually shifted

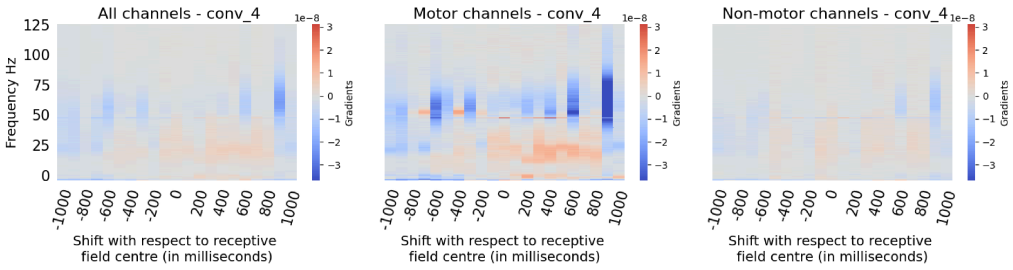
Velocity - Full dataset



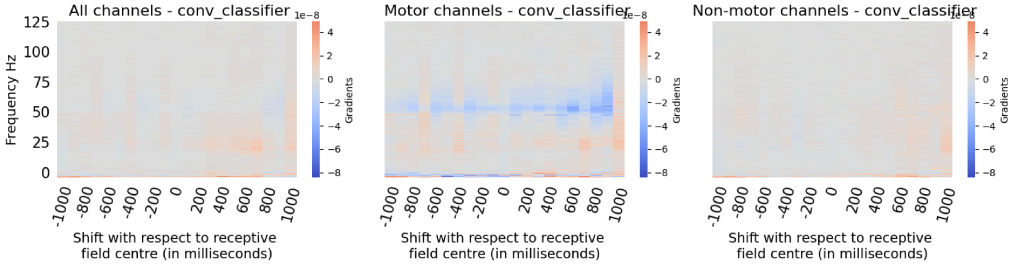
(a)



(b)



(c)



(d)

Figure C.1 Gradients of the different CNN architectures decoding velocity from the full dataset when gradually shifting the predicted time-point (see Section 4.2.3. **(a)** shows gradients of the convolutional layer in the second block; **(b)** shows gradients of the convolutional layer in the third block; **(c)** shows gradients of the fourth convolutional block; **(d)** shows gradients of the last convolutional layer - the output layer. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4

Velocity - High-passed dataset

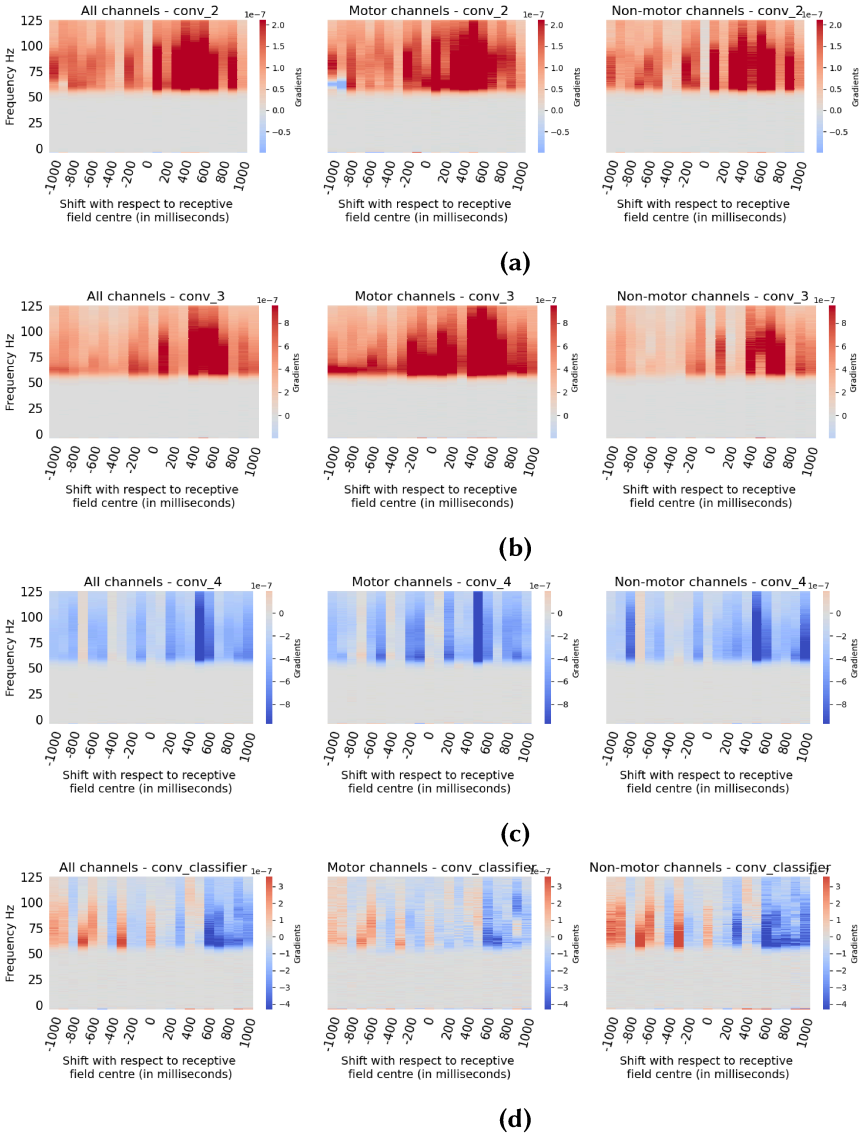


Figure C.2 Gradients of the different CNN architectures decoding velocity from the high-passed dataset when gradually shifting the predicted time-point (see Section 4.2.3. **(a)** shows gradients of the convolutional layer in the second block; **(b)** shows gradients of the convolutional layer in the third block; **(c)** shows gradients of the fourth convolutional block; **(d)** shows gradients of the last convolutional layer - the output layer. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4

Absolute velocity - Full dataset

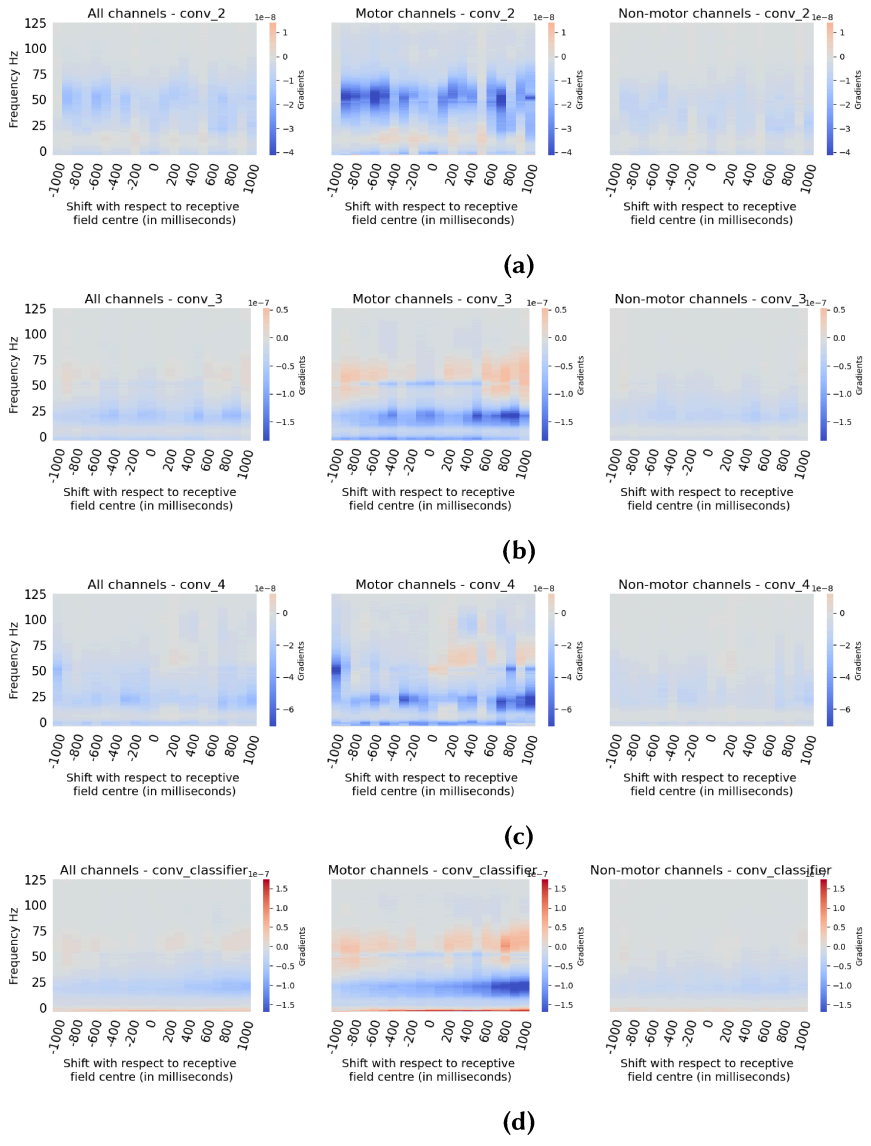


Figure C.3 Gradients of the different CNN architectures decoding absolute velocity from the full dataset when gradually shifting the predicted time-point (see Section 4.2.3). **(a)** shows gradients of the convolutional layer in the second block; **(b)** shows gradients of the convolutional layer in the third block; **(c)** shows gradients of the fourth convolutional block; **(d)** shows gradients of the last convolutional layer - the output layer. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4

Absolute velocity - High-passed dataset

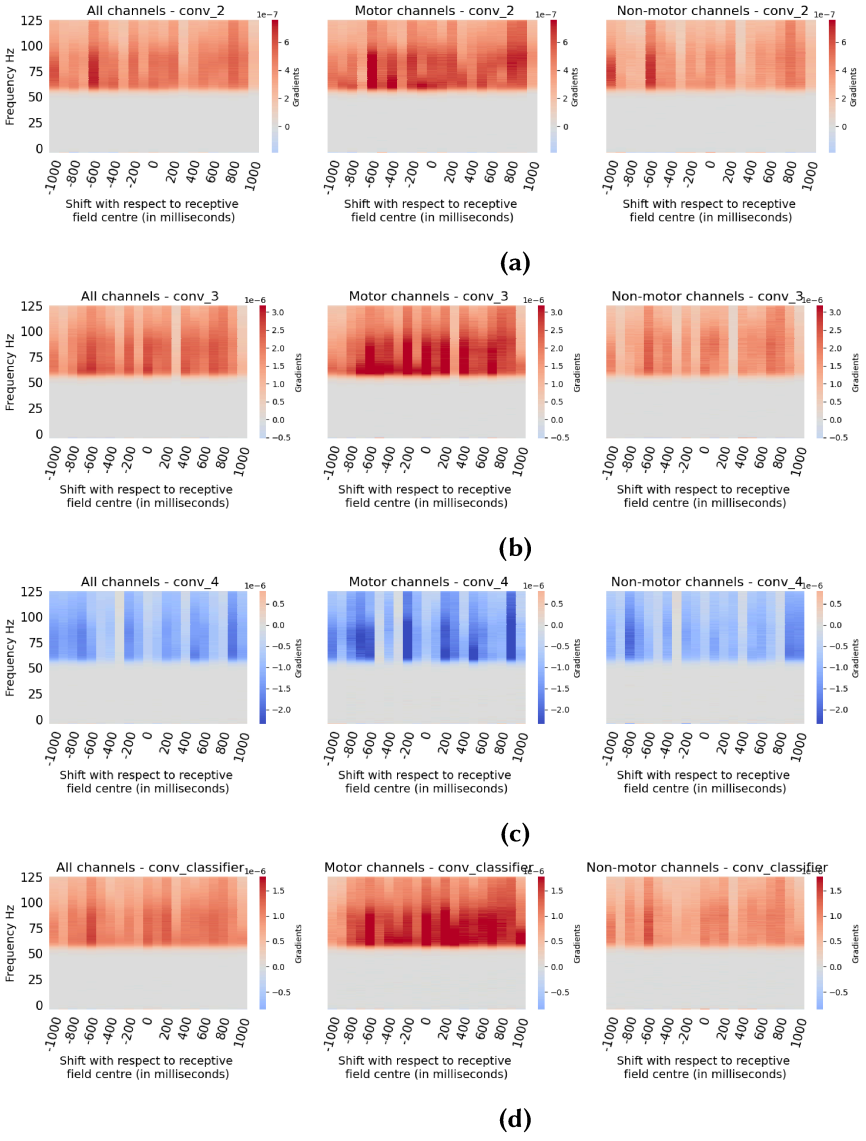
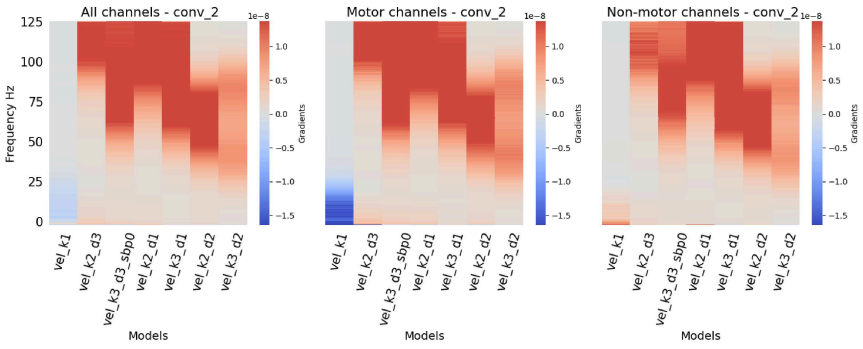


Figure C.4 Gradients of the different CNN architectures decoding absolute velocity from the high-passed dataset when gradually shifting the predicted time-point (see Section 4.2.3). **(a)** shows gradients of the convolutional layer in the second block; **(b)** shows gradients of the convolutional layer in the third block; **(c)** shows gradients of the fourth convolutional block; **(d)** shows gradients of the last convolutional layer - the output layer. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4

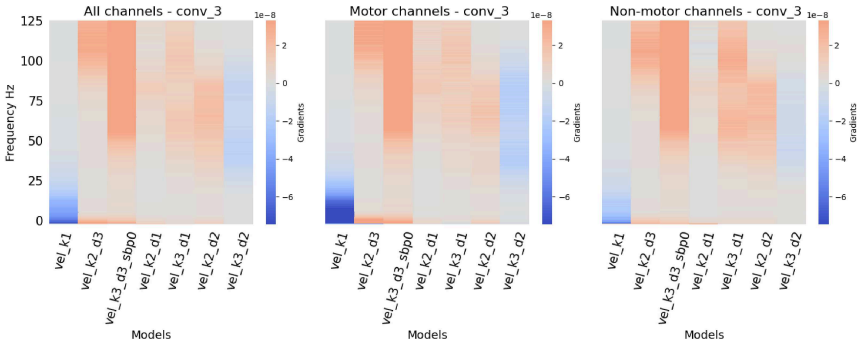
Appendix D

Gradients of all architectures - spectral whitening

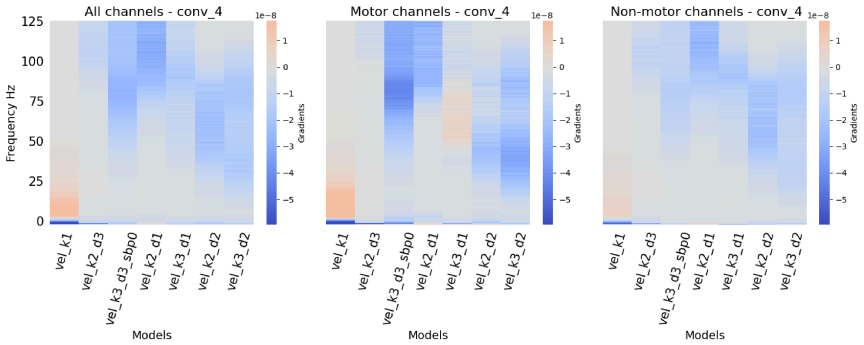
Velocity - Full dataset



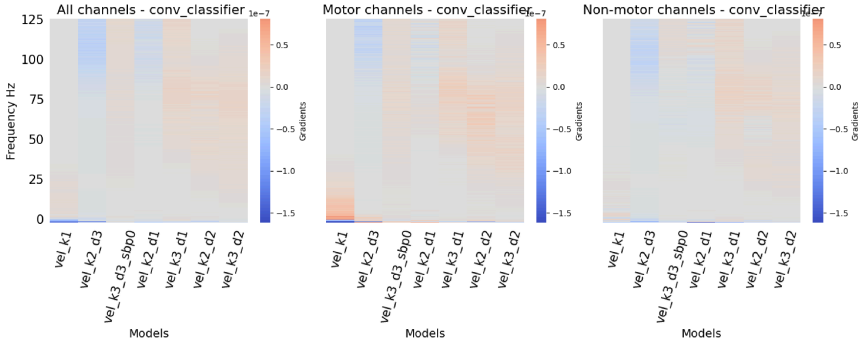
(a)



(b)



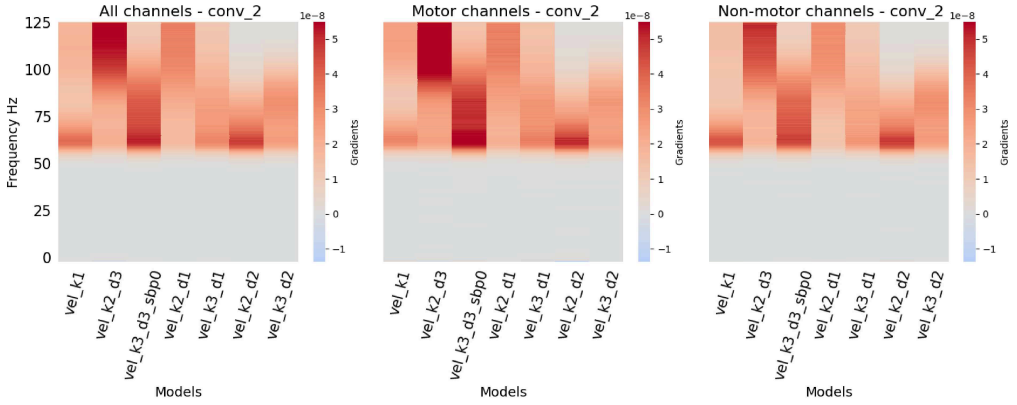
(c)



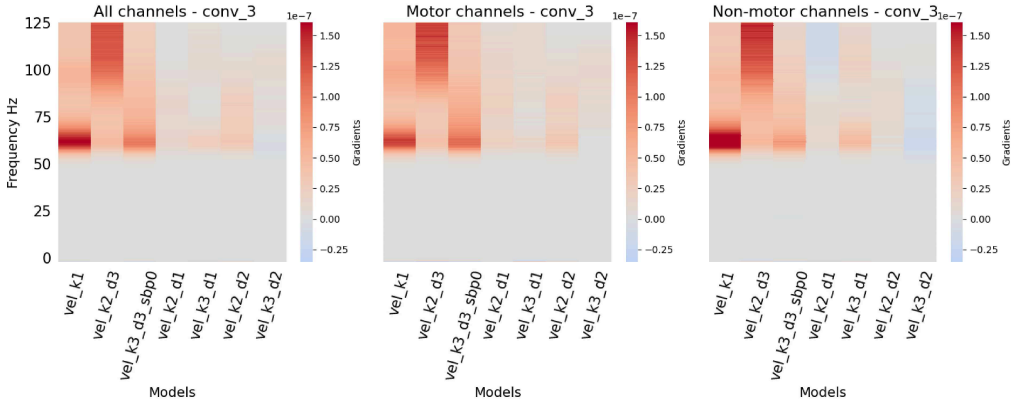
(d)

Figure D.1 Gradients of the different CNN architectures decoding velocity from the full whitened dataset in the original non-shifted setting (causal prediction) (see Section 4.3). **(a)** shows gradients of the convolutional layer in the second block; **(b)** shows gradients of the convolutional layer in the third block; **(c)** shows gradients of the fourth convolutional block; **(d)** shows gradients of the last convolutional layer - the output layer. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4

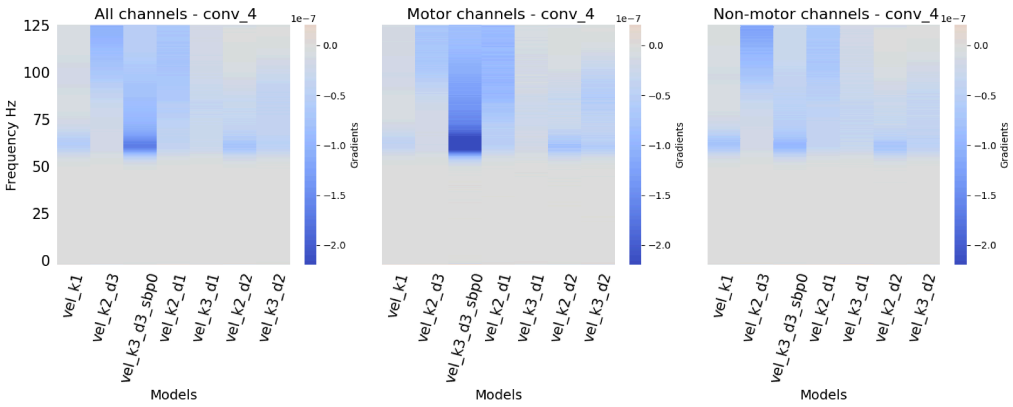
Absolute velocity - High-passed dataset



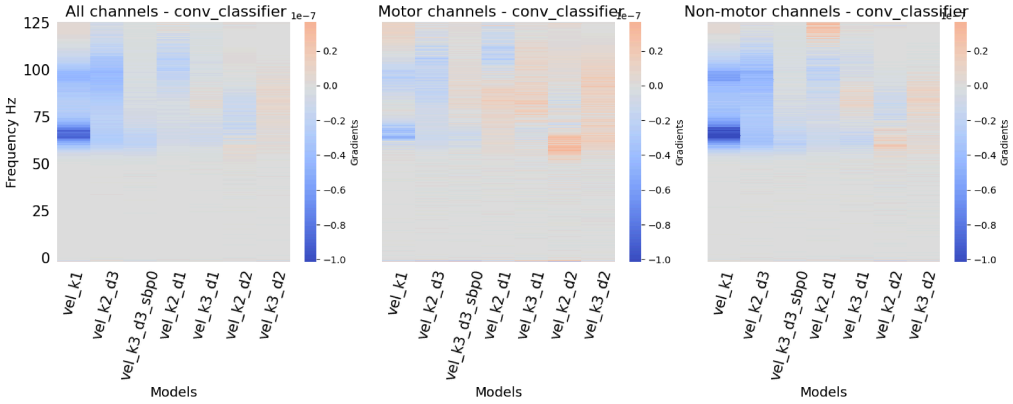
(a)



(b)



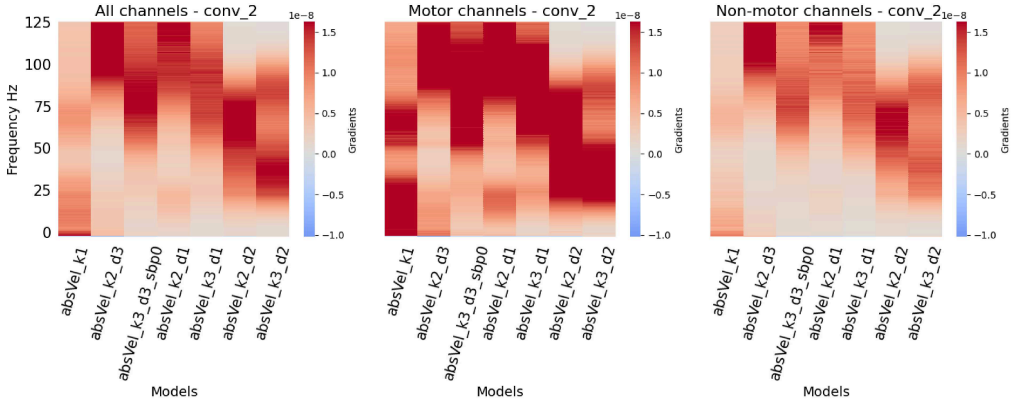
(c)



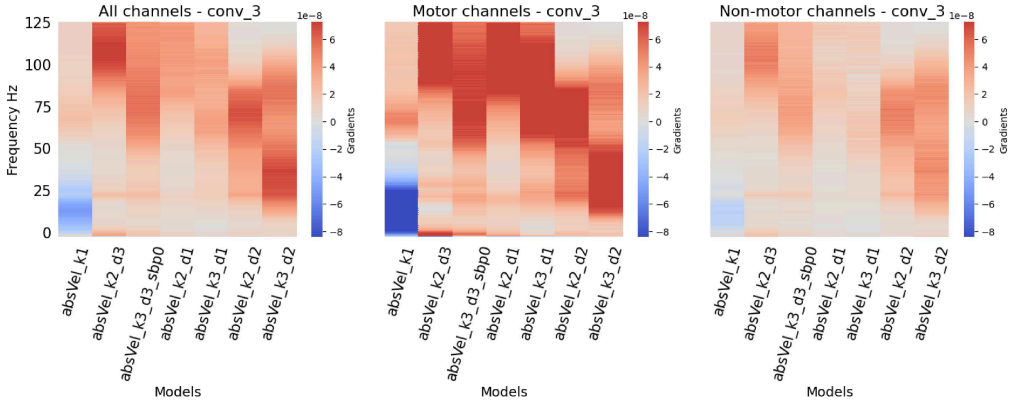
(d)

Figure D.2 Gradients of the different CNN architectures decoding velocity from the high-passed, whitened dataset in the original non-shifted setting (causal prediction) (see Section 4.3). **(a)** shows gradients of the convolutional layer in the second block; **(b)** shows gradients of the convolutional layer in the third block; **(c)** shows gradients of the fourth convolutional block; **(d)** shows gradients of the last convolutional layer - the output layer. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4

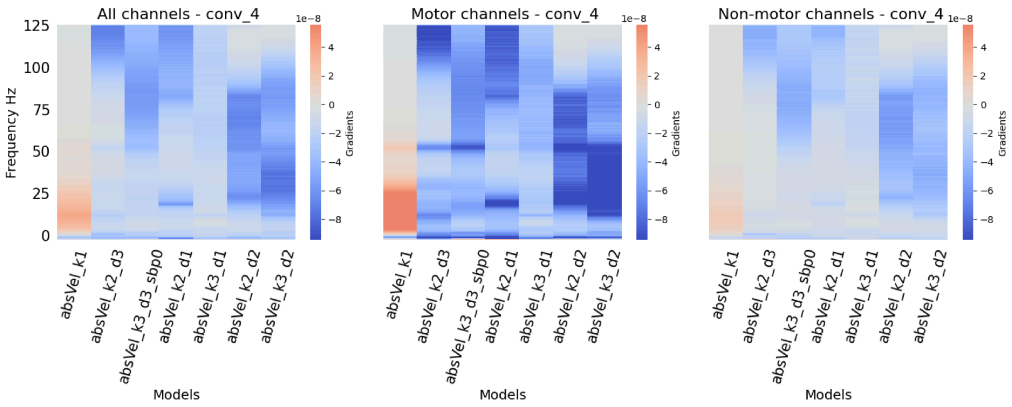
Absolute velocity - Full dataset



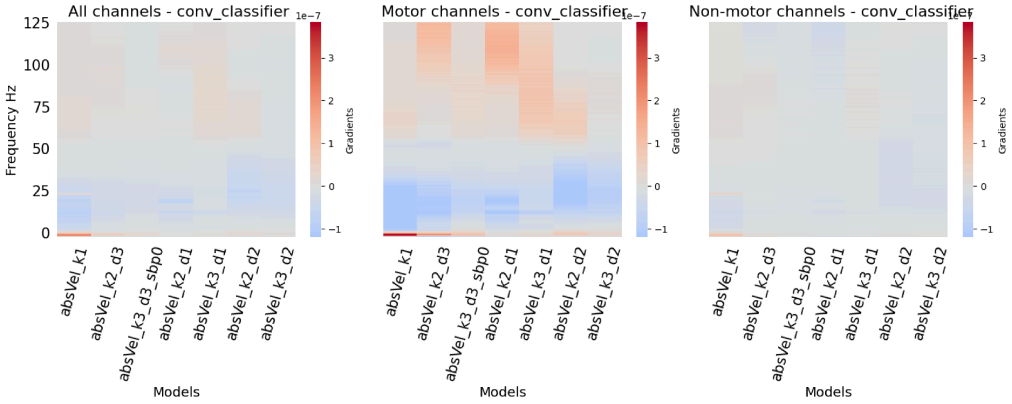
(a)



(b)



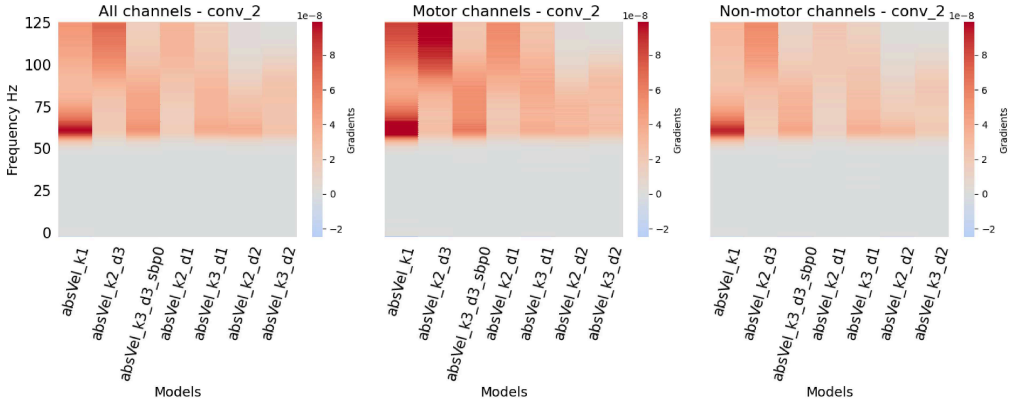
(c)



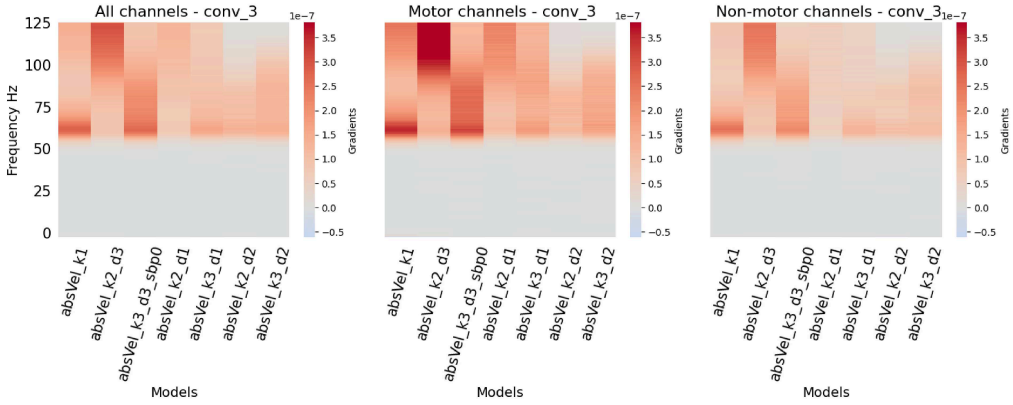
(d)

Figure D.3 Gradients of the different CNN architectures decoding absolute velocity from the full whitened dataset in the original non-shifted setting (causal prediction) (see Section 4.3). **(a)** shows gradients of the convolutional layer in the second block; **(b)** shows gradients of the convolutional layer in the third block; **(c)** shows gradients of the fourth convolutional block; **(d)** shows gradients of the last convolutional layer - the output layer. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4

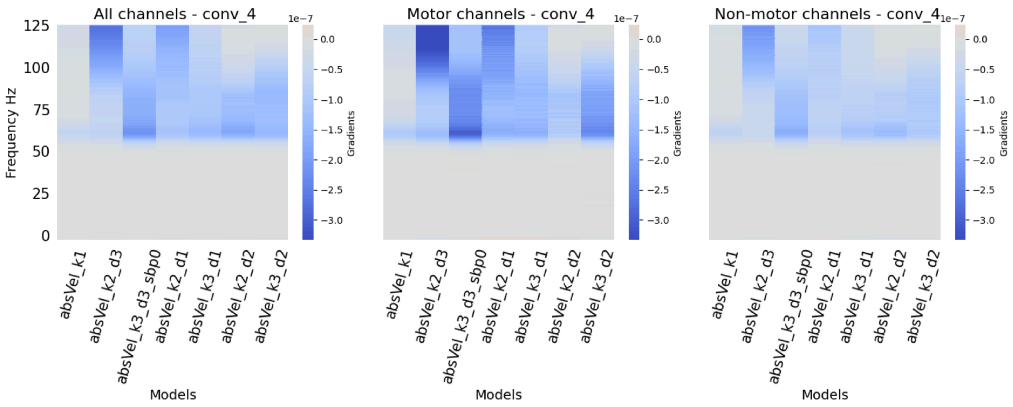
Absolute velocity - High-passed dataset



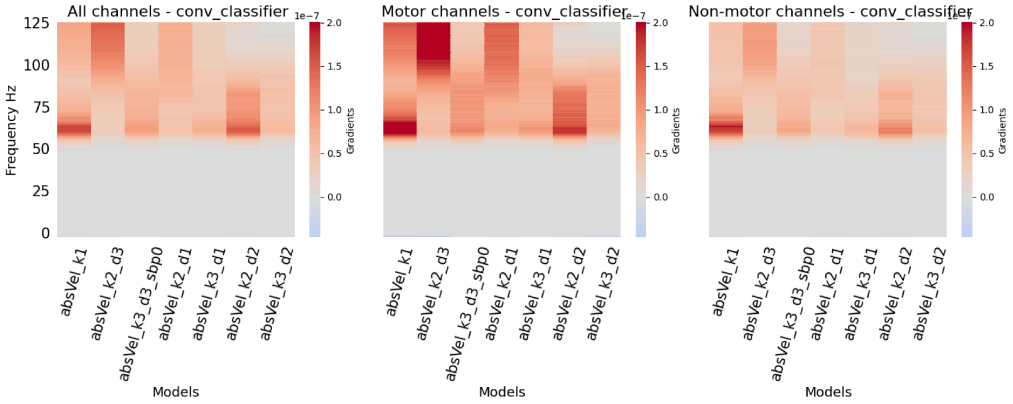
(a)



(b)



(c)



(d)

Figure D.4 Gradients of the different CNN architectures decoding absolute velocity from the high-passed whitened dataset in the original non-shifted setting (causal prediction) (see Section 4.3). **(a)** shows gradients of the convolutional layer in the second block; **(b)** shows gradients of the convolutional layer in the third block; **(c)** shows gradients of the fourth convolutional block; **(d)** shows gradients of the last convolutional layer - the output layer. All channels include channels that do not belong to motor neither non-motor channel sets. See Section 3.1.4