Alma Mater Studiorum - Università di Bologna

DOTTORATO DI RICERCA IN

INGEGNERIA BIOMEDICA, ELETTRICA E DEI SISTEMI

Ciclo 34

Settore Concorsuale: 09/G2 - BIOINGEGNERIA

Settore Scientifico Disciplinare: ING-INF/06 - BIOINGEGNERIA ELETTRONICA E INFORMATICA

INTERPRETABLE CONVOLUTIONAL NEURAL NETWORKS FOR DECODING AND ANALYZING NEURAL TIME SERIES DATA

Presentata da: Davide Borra

Coordinatore Dottorato

Michele Monaci

Supervisore

Elisa Magosso

Co-supervisore

Silvia Fantozzi

Esame finale anno 2022

This page was intentionally left blank

Table of Contents

This page was intentionally left blank

ABSTRACT

Machine learning (ML) algorithms are widely adopted to decode neural time series, including electroencephalographic (EEG) and single-cell recordings. Recent solutions based on deep learning (DL), a branch of ML, outperformed traditional ML decoders by automatically extracting relevant discriminative features from raw or minimally pre-processed neural signals, instead of using hand-crafted features. The automatic feature learning of DL decoders could be exploited also to analyze neural time series in a data-driven way, realizing novel analysis tools without adopting a priori assumptions about the underlying neural processes, helping to validate and inform cognitive neuroscience knowledge.

DL algorithms include Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs) and Fully-Connected Neural Networks (FCNNs). Among these, CNNs, originally used to classify images, have been successfully translated to the EEG and represent the most common DL-based EEG decoders adopted in the state-of-the-art (SOA), obtaining high performance. However, the current research on CNN-based decoding of neural time series is affected by some limitations. First, SOA CNNs for EEG decoding usually exploit deep and heavy structures and, thus, introduce a large number of trainable parameters (even >100K) with the risk of overfitting small datasets. The CNN architecture (e.g., the number of layers, the size and number of convolutional filters per layer) is often defined empirically, without being guided by objective and informed choices. Second, CNNs are mainly validated by designing within-subject decoders. That is, the SOA scarcely explore other training strategies such as transfer learning, which could reduce training times promoting the application of CNNs in practice (e.g., in brain-computer interfaces). In addition, the features automatically learned by CNNs remain mainly unexplored in the SOA; conversely, the interpretation of the learned features may be of great value to use decoders also as analysis tools, highlighting neural signatures underlying the different decoded brain or behavioral states in a data-driven way, without adopting many a priori assumptions about the underlying neural processes. Lastly, the SOA DL-based algorithms proposed to decode neurons' spiking rate from single-cell recordings rely on RNNs and FCNNs, which are more complex, slower to be trained and less interpretable than CNNs, and the use of CNNs for decoding this type of neural signals has not been investigated.

This PhD work addresses the previous limitations, with reference to the following decoding problems: P300 decoding from EEG, motor decoding from EEG, and motor decoding from neurons' spiking rate. In this research, CNNs were developed with particular care in the design of their architecture, keeping them light and compact (i.e., with ≤ 5 trainable layers and few thousand of trainable parameters), and adopting specific solutions to increase the interpretability of the learned features. Moreover, in this work, CNNs were validated under multiple training strategies, including also transfer learning. The proposed CNNs proved to outperform deeper, heavier, and non-interpretable solutions (e.g., with an average accuracy improvement of 22.3% and 8% while decoding motor imagery and the P300, respectively), and transfer learning resulted beneficial as it allowed reducing training times still achieving high performance (e.g., achieving accuracies > 80% in P300 decoding with only 160 training examples). Furthermore, by incorporating interpretable components into CNNs, and by applying ad-hoc techniques to interpret the learned features, CNN-based EEG analyses were

proposed to study neural features in the spatial, temporal and frequency domains, and proved to better highlight and enhance relevant neural features related to P300 and motor states than canonical EEG analyses. Remarkably, these data-driven analyses could be used, in perspective, to design novel EEG biomarkers for neurological or neurodevelopmental disorders. Lastly, CNNs were developed and applied to decode neurons' spiking rate in alternative to other DL-based algorithms, providing a better compromise between decoding performance and model complexity.

INTRODUCTION

NEURAL DECODING AND OBJECTIVE OF THE PHD RESEARCH

Neural decoding uses signals from the brain to make predictions about behavior, perception, or cognition, and it is an important field of research at the interface between engineering and neuroscience. Each neural decoding problem deals with neural activity recorded in multiple sites over a certain period of time, such as BOLD signals in brain voxels (fMRI signals), or electric potentials at multiple spatially-distributed electrodes placed on the scalp (electroencephalography signals, EEG) or implanted on the cortical surface (electrocorticography signals), or neuron's spiking rates obtained via single-cell recordings. Whatever the source of brain signals, the aim is to infer the values of observable variables, i.e., behavior (motor or verbal output) or applied stimuli (eliciting different perceptual, attentive, affective states or different movement intentions), from the recorded brain activity. Depending on the cases, the variables to be decoded can be either continuous (e.g., movement trajectory or velocity) or categorical (e.g., perceived stimulus category); in the first case the neural decoder will perform regression and in the second case it will perform classification. Two main goals can be identified as common to neural decoding: i) design brain-computer interfaces (BCIs), where the neural activity is translated into commands used to provide feedback to the user and to control an output external device (e.g., a cursor or a wheelchair); ii) help clarifying the neural representations of sensory features or movements, and to gain a better understanding of the link between neural activity and behavioral, cognitive, perceptual states. Recently, advancements in Artificial Intelligence (AI) techniques, especially in Machine Learning (ML, a branch of AI focused on applications that learn knowledge from the data), has led to a growing interest in applying these techniques to neural decoding. Indeed, the use of modern ML algorithms for neural decoding, in particular Deep Learning (DL, a branch of ML that uses artificial neural networks inspired by the human brain) has the potential to significantly improve the decoding performance and may also boost a better comprehension of neural functions (see Figure 1 about AI approaches).

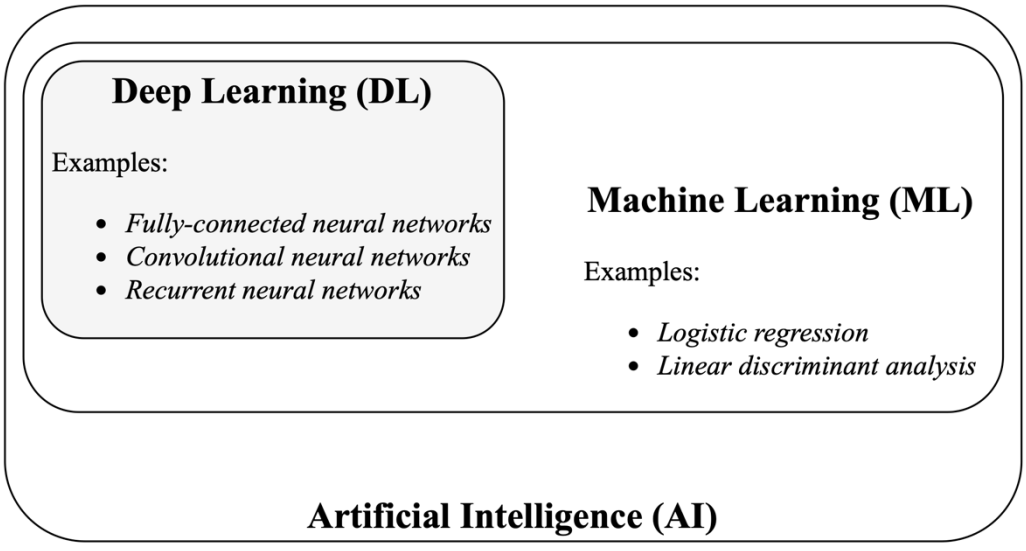


Figure 1 – Approaches of artificial intelligence.

The objective of this PhD Thesis was the development of novel DL algorithms, based on Convolutional Neural Networks (CNNs), for decoding neural time series with reference to three main decoding problems:

i. Decoding of the P300 response from EEG signals. P300 is an attention-dependent event-related potential that indices high-order selective attention and is usually elicited when a relevant target stimulus is presented inside a stream of irrelevant non-target stimuli. In this case, classification problems were addressed, and CNNs discriminated among the class of the P300-eliciting target stimulus and the classes of the other stimuli.
ii. Decoding of imagined and executed movements from EEG signals. In this context, both classification problems (decoding the body part that was moved or imagined to be moved) and regression problems (decoding the arm trajectory in a tracking task) were addressed.
iii. Decoding of executed movements from neurons' spiking rates obtained from single-cell recordings in monkeys while performing an arm-reaching task to assigned points in space. In this case too, both a classification problem (decoding the end-point of the reaching movement) and regression problem (decoding the movement trajectory) were addressed.

The proposed algorithms aspire to provide methodological improvements in CNN-based decoding of neural time series, contributing to overcome some limitations that affect the current scientific research on this topic. These improvements can have relevant implications both for increasing neuroscience knowledge, as the proposed algorithms may serve as analysis tools of neural activities, and also, in perspective, for supporting the advancement in brain-computer communication. The focus on P300 decoding and motor decoding has the following main motivations. First, detections of the P300 response and of sensorimotor rhythms or movement-related potentials (representing the main motor correlates) are widely adopted in BCIs for medical applications (e.g., rehabilitation) [1,2]. In addition, alterations in these neural correlates occur in a variety of neurological and neurodevelopmental disorders, e.g., autism, schizophrenia, depression for the P300 response [3,4], or stroke [5] for motor correlates. Thus, neural decoding, in particular via DL algorithms as presented in the following, may help improving our comprehension of these correlates.

In the next section, DL and especially CNNs are introduced in more depth. Then the state-of-the-art limitations in CNN-based decoding are presented and the improvements proposed in this Thesis to overcome these limits are illustrated. Finally, the structure of the Thesis is described in detail.

MACHINE LEARNING AND DEEP LEARNING VIA CONVOLUTIONAL NEURAL NETWORKS

A ML algorithm is an algorithm capable of learning from data, and Tom Mitchell in 1997 [6] provided a concise definition: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E". The algorithm learns from a collection of examples (training examples) to solve the target decoding problem during a training stage, and then it is evaluated on a separate collection of unseen examples (test examples). Approaches

based on ML are widely used to decode brain multi-variate activity, as recorded simultaneously from multiple sites (differently spatially distributed) using electrodes placed either on the scalp - by means of the EEG recordings - or on the cortex - by means of neurons' spiking rate via single-cell recordings.

The application of ML algorithms mainly regards the implementation of decoders for BCIs, realizing a sequence of steps that transform the input neural time series into an output decision, which serves to control an external device and to provide user feedback. Classic ML techniques operate on highly pre-processed input neural time series [7], e.g., by removing artefactual components, such as blinking and muscular artefacts, by considering only frequency components in pre-defined bands, or by considering only a selection of signals at some spatial locations (e.g., only parietal or centro-parietal electrodes for the decoding of P300). In this way, already at the pre-processing stage some a priori assumptions about the relevant neural features encoding the variables of interest are applied. Furthermore, classic ML algorithms are based on the composition of different separated stages, which are accomplished by different algorithms. The first stages perform feature extraction and selection [8], by extracting and selecting the meaningful features from the pre-processed time series; these are followed by a stage performing classification or regression [9], that provides the target decision (e.g., movement condition) on the basis of the selected features. The feature extraction stage relies on a priori assumptions about the relevant features of the neural time series to be processed. For example, in case of P300 decoding, features can be extracted in the temporal domain based on the moving-average within windows of pre-defined length and stride, assuming that these are temporal windows of relevance for the P300, or features can be extracted in the time-frequency domain based on the continuous wavelet transform within frequency bands assumed of relevance for the P300, such as delta and theta bands [10]. That is, pre-processing and feature extraction stages are guided in their implementation by a priori assumptions based on some known characteristics of the neural time series [11–13], discarding information assumed irrelevant. In this way, not all information contained in the recorded data is explored by the algorithm, preventing the algorithm to exploit also other useful information to perform the decoding; thus, features potentially relevant but so-far unknown may be discarded in these stages, and this could also negatively affect the algorithm performance.

To overcome this limitation, Deep Learning (DL) algorithms can be used. These consist in deep neural networks realized by stacking layers of artificial neurons. Depending on the established connections between these neurons, feed-forward or recurrent neural networks (RNNs), with both feed-forward and feed-back connections, can be designed. Furthermore, among feed-forward neural networks, fully-connected (FCNNs) and convolutional neural networks (CNNs) can be distinguished. All deep neural networks are characterized by a set of trainable parameters and a set of hyper-parameters.

The set of trainable parameters is composed by the weights characterizing the connections between neurons in the architecture and need to be optimized during the training stage. That is, the knowledge learned by the network during training is embedded in the set of trainable parameters. In particular, the training stage of deep neural networks consists in finding the trainable parameters that minimize a loss function (which depends on the target decoding problem, e.g., mean squared error for regression), by using gradient descent-based algorithms

and backpropagation to compute the gradient of the loss with respect to the trainable parameters.

Conversely, the hyper-parameters are parameters that define the structure of the neural network, e.g., the number of layers, the number of neurons per layer, the number of convolutional filters (see below for CNNs), and that define the training stage, e.g., the learning rate, the number of training epochs, and the optimization algorithm. These hyper-parameters need to be set before the training starts and are not known a priori. Thus, their definition usually requires an extensive and time-consuming evaluation procedure e.g., hyper-parameters can be defined empirically, by evaluating the effect of changing one hyper-parameter at a time. It is worth noticing that the evaluation of the hyper-parameters requires the use of validation examples different from the training examples and test examples, as the first are used to optimize the trainable parameters and the second are used to evaluate the performance of the trained algorithm on unseen data.

Among deep neural networks, CNNs were recently transposed to decode the EEG [14] as they require less trainable parameters and are more efficient to train respect to FCNNs and RNNs. Figure 2 shows a representative schematization of a CNN applied to neural time series. CNNs are feed-forward neural networks composed by the sequence of many layers of neurons, i.e., processing units (represented as blue circles in Figure 2), performing the convolution operation at least in one layer within the architecture (called "convolutional layer"), and followed by one or more fully-connected layers (in Figure 2 only one fully-connected layer is reported for brevity). In a fully-connected layer, all possible connections between the local input and local output neurons (where "local" refers to a specific layer inside the architecture) are realized, each one associated to a weight (see yellow connections in Figure 2). A convolutional layer performs the convolution between its local input and a set of trainable convolutional filters (see green boxes in Figure 2), providing as output a set of filtered versions of the local input, each one called "feature map". During the training stage, the weights of the fully-connected connections and the coefficient values of each filter used in each convolutional layer are learned.

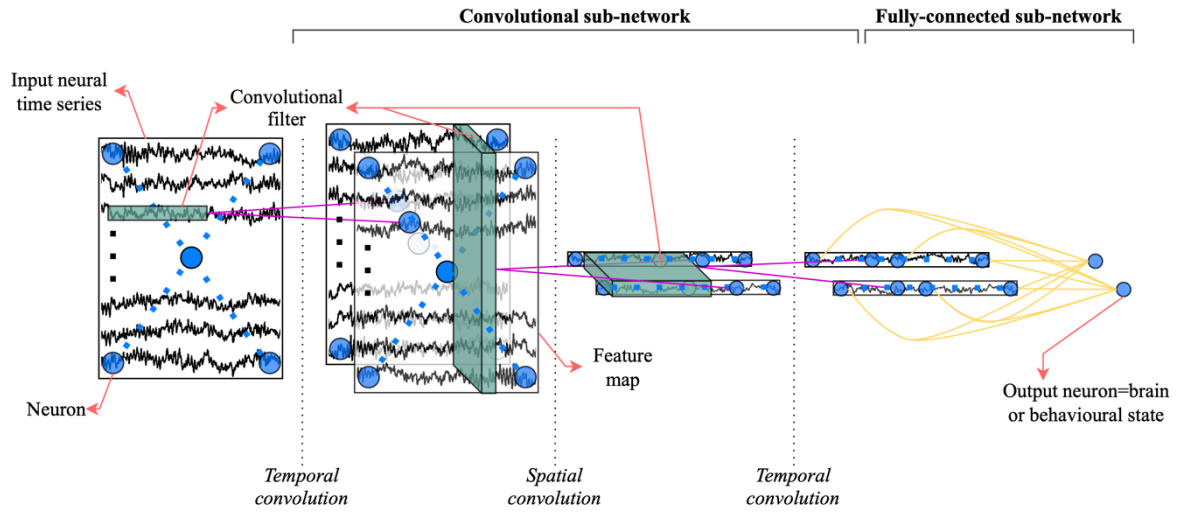


Figure 2 – Representative structure of a CNN applied to neural time series. In this example, the CNN structure is composed by the sequence of 3 convolutional layers and one fully-connected layer, representing the output layer. The neural time series is represented as a 2-D matrix with spatial locations by rows and time samples by columns.

Neurons are represented as blue circles, while convolutional filters are represented by green boxes. Note that, for brevity, only the first and last feature maps are displayed. Purple and yellow connections denote convolutional and fully-connected connections, respectively.

At variance with the fully-connected layers, the connections between the local input and local output of a convolutional layer are sparse by design, as each output neuron of one feature map takes connections only with the subset of input neurons that falls within the convolutional filter (see purple connections in Figure 2). This provides a reduction of trainable parameters and a reduction of computation time (less operations to perform). Lastly, as each convolutional filter is reused at every position of the local input (possibly except for boundary elements, depending on the padding of the convolution) the parameters of a convolutional layer are shared across different locations of the local input, further reducing the number of parameters to optimize [15].

Each neuron belonging to a feature map of a convolutional layer responds to features occurring within the convolutional filter size, and thus, is characterized by a local receptive field corresponding to the convolutional filter size. By design, the CNN structure is inspired to the hierarchical structure of the ventral stream of the visual system. Indeed, by stacking neurons with specific local receptive fields on top of others, global receptive fields of neurons increase with the network depth and the learned features (i.e., the set of convolutional filters) increase in complexity and abstraction, i.e., simpler and less abstract features are learned in the first layers, while more complex and abstract features are learned in the last layers [16].

CNNs can be viewed as computational (or "functional") models [17] that learn to approximate the transformation to be applied to the neural time series to provide the corresponding brain or behavioral state (e.g., movement condition) as output, automatically learning useful features to realize this input-output mapping. That is, when applying these convolutional models to neural time series, the model input is represented by the multi-variate brain activity and the model outputs are represented by the states to be decoded. A classic topology of these models, as represented in Figure 2, is defined by the composition of an input layer, composed by neurons that simply replicate the brain activity, a convolutional sub-network, including only convolutional layers as trainable layers, and a fully-connected sub-network, including one or more fully-connected layers as trainable layers. The last fully-connected layer (corresponding to the output layer) comprises one neuron for each output condition to be predicted, e.g., 2 output neurons if we are interested in predicting 2 different movement conditions. The convolutional sub-network is devoted to learning an encoded and more convenient representation of the input multi-variate brain activity for the decoding problem, exploiting only convolutional features. The fully-connected sub-network is devoted to processing the representation provided by the previous convolutional sub-network, to associate the correct behavioral or brain state to the input, i.e., it finalizes the decoding problem.

Contrary to classic ML algorithms, in CNN-based algorithms there is no separation between the feature extraction, feature selection and classification (or regression) stages: given a multi-variate neural time series as input, the decoded decision is provided as output in an end-to-end fashion. Furthermore, the requirement of a high decoding-oriented pre-processing is relaxed, as the CNN can automatically learn to filter out unrelated information and focus only on the most important ones for the decoding problem, exploiting the entire spatial and temporal

information contained in the input multi-variate neural time series. Most of the studies focusing on neural decoding with deep neural networks, in particular with the EEG, exploited the entire set of electrode signals, automatically learning to focus only on a small subset of them; furthermore, simple pre-processing pipelines were employed, including broad band-pass filtering, downsampling and re-referencing [14]. Lastly, artifacts were not commonly handled, and this did not hamper decoding performance [14]. Overall, these properties give CNNs the ability to learn the relevant neural features related to the investigated brain or behavioral states (end-to-end automatic feature learning), from raw or minimally pre-processed multi-variate neural time series, without exploiting a priori assumptions about the characteristics of the neural response. It is worth remarking that this represents an important advantage over the classic ML approaches, in which only limited characteristics of the input neural time series, assumed relevant a priori, are exploited for decoding. Crucially, this property of CNNs not only could be beneficial to improve decoding accuracy, but also to realize novel analysis tools of neural time series. Indeed, despite the knowledge learned by CNNs is not interpretable and these models are often marked as "black boxes", techniques can be used to increase the CNN interpretability and analyze the trained CNN. This could be used for determining, for example, which elements of the input or which learned features are most informative for the decoding, gaining insights about the neural correlates underling the predicted states. That is, the CNN could also represent a key component to design a data-driven analysis technique, potentially helping to validate and inform cognitive neuroscience knowledge.

LIMITATIONS OF THE STATE-OF-THE-ART AND THE PROPOSED APPROACHES TO OVERCOME THEM

In the following, the main limitations of the state-of-the-art of CNN-based neural decoding, regarding the network architecture, training strategy, feature interpretation and decoding of neurons' spiking rate, are presented together with the approaches proposed in the present Thesis to overcome them.

Network architecture (or structure)

In literature, most of the approaches based on CNNs to decode the EEG arrange the input neural time series (and, thus, the input layer of the model) as a 2-D matrix (spatial locations by rows and time steps by columns) [14], preserving the original input representation [18–26]. Then, first convolutions of the convolutional sub-network separately operate on the different domains of the input time series. That is, they operate separately in the temporal domain (temporal convolution) – filtering in time each time series – and in the spatial domain (spatial convolution) [18–21,23–26] – recombining the time series across spatial locations (see Figure 2). Less frequent solutions operate simultaneously on both spatial and temporal domains (mixed spatio-temporal convolution) [22,27]. Later convolutions generally continue learning more complex and abstract features in the temporal domain. Lastly, generally only one fully-connected layer, corresponding to the output layer, is used in the fully-connected sub-network. SOA convolutional models differ in the sequence and type (mixing or not the learning across domains) of the first convolutions operating directly on the input spatial and temporal domains, and in the number of deeper layers learning more complex temporal features. Despite this

common CNN structure across studies [18–21,23–26], many others hyper-parameters of the model largely differ across different decoders, and the justification for one choice rather than another is unclear and not properly investigated. These hyper-parameters include, for example, the number of convolutional filters and the length of these filters of the first two convolutions operating in the temporal and spatial domains, and the design of later temporal convolutions. Furthermore, as other DL models, in general CNNs are data-hungry algorithms requiring large datasets to tune their trainable parameters. However, CNNs can be designed to realize accurate decoders for BCI applications with relatively compact EEG datasets, e.g., using from 10 to 1000 examples/minute of the recorded EEG (most frequent value: approx. 80 examples/minute of recording) and recording up to 100 subjects (most frequent value: approx. 10 subjects) [14]. Due to the limited size of EEG datasets (both in terms of number of examples per subject and of number of subjects), the model size, defined by the number of trainable parameters of the decoder, should be carefully designed by keeping limited the number of trainable parameters to improve generalization. SOA convolutional models differ in the adoption of a heavy or light and parsimonious architecture, thus, SOA models present a high variability in the number of trainable parameters introduced. Only few models [20,22] focus on the design of a parsimonious model (e.g., including ~2K parameters), while most models consist in deep and heavy algorithms (e.g., including ~300K parameters in the network proposed by Schirrmeister et al. [24]) that could not generalize well on more compact and limited datasets compared to the ones they were tested on.

Therefore, in this Thesis these limitations related to the CNN architecture were addressed by:

i. Investigating the optimal architecture by evaluating alternative structures obtained by changing one hyper-parameter at a time (post-hoc hyper-parameter evaluations) or by searching the optimal architecture using an algorithm that performs automatic hyper-parameter search.
ii. Adopting models that are not excessively deep (i.e., compact models) and that widely use specialized convolutions aimed to reduce the number of trainable parameters compared to traditional convolutions.

Both these solutions allowed to keep limited the overall number of trainable parameters of the proposed architectures, promoting the design of parsimonious models.

Training strategy

In the literature, only few strategies are adopted to train the decoders, mainly training subject-specific decoders using subject-specific training examples [18,19,21,22,24–26,28], i.e., within-subject training strategy. This could be limiting as, for a more complete evaluation, decoders should be evaluated using multiple training strategies, where each training strategy could reflect a different BCI practical scenario. For example, it may be useful to assess how a CNN trained on data from a group of subjects performs on data from a new subject; this may represent the case of a new participant approaching a BCI intervention, and a CNN already trained on the previous participants can be directly used on the new one, preventing to waste time for a new calibration (training) stage.

Therefore, in this Thesis decoders were widely evaluated by adopting different training strategies. In particular, when the adopted datasets consisted in multiple recording sessions per subject, within-session and cross-session training strategies were analyzed. These consisted in training and evaluating decoders with signals collected during one single session, or multiple recording sessions. Furthermore, also a cross-subject training strategy was evaluated, where signals from multiple subjects were used to train decoders and signals from a new subject were used to test decoders. Remarkably, it was also investigated the potentiality and the benefit of transferring the knowledge from decoders trained on other subjects to a new one (transfer learning). In this case, models were trained on the new participant but, instead of using a random initialization of the trainable parameters, models were initialized with the trainable parameters learned from other participants, enabling a performance improvement even with fewer training examples, thus, reducing the calibration time of BCIs and promoting the use of CNNs in practice.

Feature interpretation

Feature interpretation means interpreting and understanding the knowledge learned by the CNN and embedded in its trainable parameters. In most cases, the features are not directly interpretable (or they are not in a form that can be easily understood) and specific techniques, called "post-hoc interpretability analyses", must be applied for their interpretation. A CNN can also be made directly interpretable in its design, by inserting components learning features that are immediately understandable, and thus, realizing an "ad-hoc interpretable model". It is worth noticing that designing an ad-hoc interpretable model generally decreases the number of trainable parameters of the model, i.e., it not only increases the interpretability but also promotes a light structure.

When performing EEG decoding, efforts have been made in the literature to interpret models by mainly applying post-hoc interpretability analyses with the purpose of verifying that the CNN-based decoders rely on neurophysiological features and not on artefactual features. Among these methodologies, convolutional filters of layers operating in the easiest interpretable domain, such as the spatial domain (each spatial location being weighted more or less depending on its importance), can be visualized [18–21,26], see Figure 3 for a representative example.

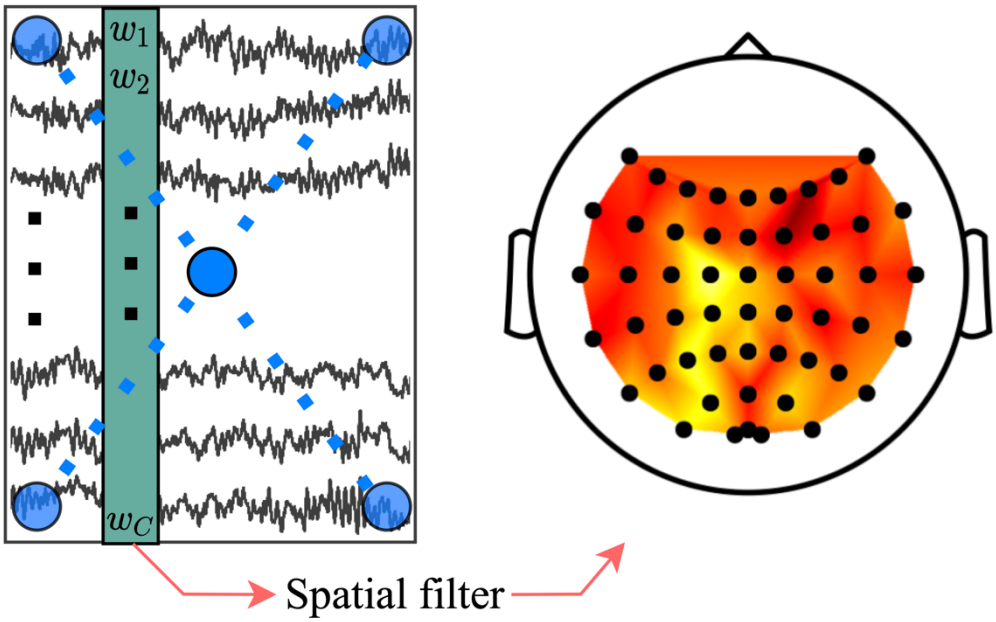


Figure 3 – Representative example of the visualization of one spatial filter learned in spatial convolution while decoding EEG signals recorded from C spatial locations. The local input for the spatial convolution (represented on the left) is extracted from the CNN reported in Figure 2. Each spatial filter is composed by C weights, one for each spatial location. Once trained the CNN, the spatial filters can be visualized as heatmaps at the level of scalp, by mapping each spatial location to its weight. Higher coefficients of the spatial filter (high values are denoted with yellow in the figure) denotes more important electrodes for discriminating among different brain or behavioral states (e.g., between movements of the left hand, right hand, and tongue). However, it is worth noticing that there is no association between this information and one specific state (e.g., movement of the right hand).

Conversely, convolutional filters operating in the temporal domain are not extensively analyzed as they are less interpretable. Other methodologies remove one convolutional filter at a time, analyzing how the resulting altered filtering affects the model performance (filter ablation) [20], assigning more or less importance to the removed filter depending on the performance variation.

The previous analyses focus on interpreting the learned CNN features without relating these features to a specific decoded brain or behavioral state. That is, there is not a strict relation between the so interpreted features (e.g., the spatial filter of Figure 3) and one specific output neuron corresponding to a specific decoded class (e.g., movement of left hand). Relating features separately to each output neuron requires to explain the network decision, i.e., to identify the interpretable features that contribute more to a network decision (activation of a specific output neuron) given a test example as input, by using so called "explanation techniques" [20,23,29]. Explanation techniques might produce more interesting representations than the previous techniques, as explanations are, by design, strictly related to a brain or behavioral state, enabling a straightforward analysis of the relevant features for a state under investigation. These techniques usually provide representations highlighting features (typically in the spatial and temporal domains) that contribute more to produce the network decision towards one brain or behavioral state. The simplest way to provide an explanation of the model is to quantify how much a small change in the values of a local input neuron affects the output neuron related to a brain or behavioral state to be decoded [30]. That is, a sensitivity analysis [31] is designed, providing representations of the most relevant

samples belonging to the local input that drive the decision for a decoded output; this sensitivity analysis could be performed by backpropagating through the CNN the gradient from an output neuron to local input neurons. When the local input corresponds to the input of the CNN, each local input neuron corresponds to each spatio-temporal sample of the neural time series and the designed technique explains the network decision at the level of the input ("input explanation technique").

However, these methodologies proposed in the literature have two main limitations. They are proposed only to validate the trained decoders by evaluating whether neurophysiological (and not artefactual) features are learned. Therefore, it is not fully exploited the potentiality of the automatic feature learning provided by CNNs to propose new analysis tools for neural time series aimed to analyze and compare different brain or behavioral states in a data driven way. In addition, input explanation techniques are mainly applied, as the domain of the neural time series (i.e., spatio-temporal domain) representing the CNN input, is the easiest interpretable domain. On the contrary, the subsequent layers of the CNN, in particular the convolutional layers operating in the temporal domain, are not easily interpretable; indeed, each filter coefficient of temporal convolutional filters is learned during training, resulting in general in filters that are not well-defined in type (i.e., they cannot be clearly recognized as low-pass, band-pass or high-pass filters) and, thus, the spectral features learned by the bank of filters more important for the decoding decision remain essentially unclear. This is limiting as it may be interesting to investigate the relevant features for a brain or behavioral state, not only in the spatio-temporal domain of the input, but also in other domains such as the frequency domain; these features are processed in intermediate layers of the CNN, thus, "intermediate explanation techniques" should be designed. Furthermore, if spatial filters are learned after the temporal convolution (as performed in the example reported in Figure 2), it is also possible to interpret the more relevant spatial locations associated to each frequency content.

Therefore, in this Thesis these two main limitations were addressed by:

i. Enabling an intermediate explanation to analyze features in the frequency domain. To this aim, the CNN structure needs to be designed interpretable in its temporal convolutional filters, realizing a model that incorporates the interpretability itself, adopting an ad-hoc interpretable modelling approach. That is, interpreting the model a posteriori using a post-hoc interpretation analysis is not enough, and the model needs to be renovated by inserting interpretable components directly in the architecture. In this Thesis, this was realized by designing temporal convolutions adopting trainable parameters that were directly interpretable, e.g., designing the bank of trainable temporal filters so that the cut-off frequencies can be directly learned instead of learning each coefficient of the filter. Finally, by coupling the interpretable model with an explanation technique operating at the level of the temporal convolution (which is an intermediate point of the CNN structure, see Figure 2), it was possible to design an explanation in the frequency domain.
ii. Proposing novel EEG analysis tools based on CNNs coupled with explanation techniques. These analyses were developed and applied to gain insights into neural features underlying brain or behavioral states both in the spatio-temporal domain and in the frequency domain (exploiting the methodologies applied in Section *Feature interpretation*-i).

Decoding neurons' spiking rate from single-cell recordings

In the literature, the decoding of neurons' spiking rate strongly relies on classic ML algorithms and there is a recent growing interest in the application of FCNNs and RNNs [19]. However, as stated before, these algorithms require more trainable parameters and are less efficient to train than CNNs. Thus, by exploiting the expertise deepened in designing CNNs to decode the EEG (especially in Section *Network architecture (or structure)*-i, ii), CNNs were also investigated and proposed in this Thesis to decode neurons' spiking rate, addressing both classification and regression problems.

THESIS STRUCTURE

This Thesis is structured by presenting the conducted studies without following a chronological order (i.e., not presenting the studies by publication date), but on the basis of the decoding problem addressed (P300 and motor decoding) and the types of neural time series decoded/analyzed (EEG and neurons' spiking rate). This structure was chosen with the aim of increasing its readability.

Section I is focused on P300 decoding from EEG signals and contains the following Chapters.

- **Chapter 1** reports a benchmark study on P300 decoding from the EEG, comparing different ML and DL (including CNNs) algorithms. This study was conducted in occasion of an international scientific competition (International Federation of Medical and Biological Engineering 2019 challenge held during the XV Mediterranean Conference on Medical and Biological Engineering and Computing). That is, at first a comparison analysis among these families of decoding algorithms is presented, identifying the best-performing algorithm for P300 decoding. The CNN-based decoder we adopted resulted the best-performing algorithm in the competition and was light in its structure (see Section *Network architecture (or structure)*-ii). This CNN structure represents the starting point for the investigations carried out in the following chapters of this section.
- **Chapter 2** presents a novel CNN-based analysis of P300 from the EEG by exploiting input explanations, investigating the more relevant features in the spatio-temporal domain of the input (see Section *Feature interpretation*-i, ii).
- **Chapter 3** describes a novel interpretable CNN (i.e., by inserting components that are directly interpretable) to decode the P300 response from the EEG, and its structure is automatically defined by using automatic hyper-parameter search while keeping the architecture light (see Section *Network architecture (or structure)*-i, ii). In addition, by using an intermediate explanation technique, a novel CNN-based analysis of P300 is proposed, investigating the neural features in the frequency domain (see Section *Feature interpretation*-i, ii). Lastly, different training strategies are investigated (see Section *Training strategy*) and compared.
- **Chapter 4** reports a study on the optimal architecture of a novel light CNN to decode the P300 from the EEG, with a particular focus on the design of deeper layers of the

convolutional sub-network (see Section *Network architecture (or structure)*-i, ii). The CNN was evaluated on three P300 dataset, acquired in different paradigms. In addition, the evaluated models are investigated with multiple training strategies (see Section *Training strategy*).

Section II is focused on motor decoding from EEG signals and contains the following Chapters.

- **Chapter 5** presents a comparison between CNNs and the state-of-the-art ML algorithm to decode motor imagery and motor execution conditions from the EEG. In addition, a novel light and interpretable CNN is presented, and its optimal architecture is investigated (see Section *Network architecture (or structure)*-i, ii). Then, novel CNN-based analyses of EEG correlates related to motor execution and motor imagery are conducted. These are performed by using intermediate explanations, investigating features in the frequency domain (see Section *Feature interpretation*-i, ii).
- **Chapter 6** presents a novel light and interpretable CNN realized to predict, from the EEG, the 2-D hand position and velocity components during upper-limb movements (tracking task). The proposed CNN is evaluated adopting different training strategies (see Section *Training strategy*). Then, a novel CNN-based analysis of EEG correlates related to position and velocity of executed movements is described. Here, an intermediate explanation is used, investigating the neural features in the frequency domain (see Section *Feature interpretation*-i, ii).

Section III is focused on motor decoding from neurons' spiking rate and contains the following Chapter.

- **Chapter 7** introduces the adoption of CNNs to decode neurons' spiking rate from single-cell recordings of monkeys during a 3-D reaching task (see Section *Decoding neurons' spiking rate from single-cell recordings*). The CNN is light in its design, with its structure optimized using automatic search algorithms (see Section *Network architecture (or structure)*-i, ii), and is used to predict both the reached points and the 3-D hand position, respectively.

Finally, a **Conclusions** section highlights the main results and main implications of the conducted studies together with the limitations that can be addressed in the future.

REFERENCES

[1] Abdulkader S N, Atia A and Mostafa M-S M 2015 Brain computer interfacing: Applications and challenges *Egyptian Informatics Journal* **16** 213–30

[2] Mak J N and Wolpaw J R 2009 Clinical Applications of Brain-Computer Interfaces: Current State and Future Prospects *IEEE Rev. Biomed. Eng.* **2** 187–99

[3] Cui T, Wang P P, Liu S and Zhang X 2017 P300 amplitude and latency in autism spectrum disorder: a meta-analysis *Eur Child Adolesc Psychiatry* **26** 177–90

[4] Picton T W 1992 The P300 Wave of the Human Event-Related Potential *Journal of Clinical Neurophysiology* **9** 456–79

[5] Finnigan S and van Putten M J A M 2013 EEG in ischaemic stroke: Quantitative EEG can uniquely inform (sub-)acute prognoses and clinical management *Clinical Neurophysiology* **124** 10–9

[6] Mitchell T M 2013 *Machine learning* (New York: McGraw-Hill)

[7] Bashashati A, Fatourechi M, Ward R K and Birch G E 2007 A survey of signal processing algorithms in brain–computer interfaces based on electrical brain signals *Journal of Neural Engineering* **4** R32–57

[8] McFarland D J, Anderson C W, Muller K-, Schlogl A and Krusienski D J 2006 BCI meeting 2005-workshop on BCI signal processing: feature extraction and translation *IEEE Transactions on Neural Systems and Rehabilitation Engineering* **14** 135–8

[9] Lotte F, Bougrain L, Cichocki A, Clerc M, Congedo M, Rakotomamonjy A and Yger F 2018 A review of classification algorithms for EEG-based brain–computer interfaces: a 10 year update *Journal of Neural Engineering* **15** 031005

[10] de Arancibia L, Sánchez-González P, Gómez E J, Hernando M E and Oropesa I 2020 Linear vs Nonlinear Classification of Social Joint Attention in Autism Using VR P300-Based Brain Computer Interfaces *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1869–74

[11] Ang K K, Chin Z Y, Zhang H and Guan C 2008 Filter bank common spatial pattern (FBCSP) in brain-computer interface *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)* (IEEE) pp 2390–7

[12] Farwell L A and Donchin E 1988 Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials *Electroencephalography and Clinical Neurophysiology* **70** 510–23

[13] Krusienski D J, Sellers E W, Cabestaing F, Bayoudh S, McFarland D J, Vaughan T M and Wolpaw J R 2006 A comparison of classification techniques for the P300 Speller *J. Neural Eng.* **3** 299–305

[14] Roy Y, Banville H, Albuquerque I, Gramfort A, Falk T H and Faubert J 2019 Deep learning-based electroencephalography analysis: a systematic review *Journal of Neural Engineering* **16** 051001

[15] Goodfellow I, Bengio Y and Courville A 2016 *Deep Learning* (MIT Press)

[16] Lindsay G 2020 Convolutional Neural Networks as a Model of the Visual System: Past, Present, and Future *Journal of Cognitive Neuroscience* 1–15

[17] Kay K N 2018 Principles for models of neural information processing *NeuroImage* **180** 101–9

[18] Cecotti H and Graser A 2011 Convolutional Neural Networks for P300 Detection with Application to Brain-Computer Interfaces *IEEE Transactions on Pattern Analysis and Machine Intelligence* **33** 433–45

[19] Manor R and Geva A B 2015 Convolutional Neural Network for Multi-Category Rapid Serial Visual Presentation BCI *Frontiers in Computational Neuroscience* **9** 146

[20] Lawhern V J, Solon A J, Waytowich N R, Gordon S M, Hung C P and Lance B J 2018 EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces *Journal of Neural Engineering* **15** 056013
[21] Liu M, Wu W, Gu Z, Yu Z, Qi F and Li Y 2018 Deep learning based on Batch Normalization for P300 signal detection *Neurocomputing* **275** 288–97
[22] Shan H, Liu Y and Stefanov T 2018 A Simple Convolutional Neural Network for Accurate P300 Detection and Character Spelling in Brain Computer Interface *Proceedings of the 27th International Joint Conference on Artificial Intelligence* IJCAI'18 (Stockholm, Sweden: AAAI Press) pp 1604–10
[23] Farahat A, Reichert C, Sweeney-Reed C and Hinrichs H 2019 Convolutional neural networks for decoding of covert attention focus and saliency maps for EEG feature visualization *Journal of Neural Engineering*
[24] Schirrmeister R T, Springenberg J T, Fiederer L D J, Glasstetter M, Eggensperger K, Tangermann M, Hutter F, Burgard W and Ball T 2017 Deep learning with convolutional neural networks for EEG decoding and visualization *Human brain mapping* **38** 5391–420
[25] Tang Z, Li C and Sun S 2017 Single-trial EEG classification of motor imagery using deep convolutional neural networks *Optik* **130** 11–8
[26] Zhao D, Tang F, Si B and Feng X 2019 Learning joint space–time–frequency features for EEG decoding on small labeled data *Neural Networks* **114** 67–77
[27] Sturm I, Lapuschkin S, Samek W and Müller K-R 2016 Interpretable deep neural networks for single-trial EEG classification *Journal of Neuroscience Methods* **274** 141–5
[28] Tabar Y R and Halici U 2016 A novel deep learning approach for classification of EEG motor imagery signals *Journal of Neural Engineering* **14** 016003
[29] Vahid A, Mückschel M, Stober S, Stock A-K and Beste C 2020 Applying deep learning to single-trial EEG data provides evidence for complementary theories on action control *Commun Biol* **3** 112
[30] Montavon G, Samek W and Müller K-R 2018 Methods for interpreting and understanding deep neural networks *Digital Signal Processing* **73** 1–15
[31] Simonyan K, Vedaldi A and Zisserman A 2014 Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps *arXiv:1312.6034 [cs]*

SECTION I: P300 DECODING FROM ELECTROENCEPHALOGRAPHIC SIGNALS

CHAPTER 1: COMPARISON BETWEEN CLASSIC ML AND DL ALGORITHMS

The study reported in this chapter refers to the published journal paper entitled "BCIAUT-P300: A Multi-Session and Multi-Subject Benchmark Dataset on Autism for P300-Based Brain-Computer-Interfaces" M. Simões*, D. Borra*, et al., *Frontiers in Human Neuroscience*, 2020. * These two authors share the first authorship. The first part of this chapter presents the P300 dataset that was used in occasion of the 2019 IFMBE scientific competition organized during MEDICON 2019 conference; the second part analyzes and compares ML and DL techniques to decode the P300 response.

There is a lack of multi-session P300 datasets for Brain-Computer Interfaces (BCI). Publicly available datasets are usually limited by small number of participants with few BCI sessions. In this sense, the lack of large, comprehensive datasets with various individuals and multiple sessions has limited advances in the development of more effective data processing and analysis methods for BCI systems. This is particularly evident to explore the feasibility of deep learning methods that require large datasets. Here we present the BCIAUT-P300 dataset, containing 15 autism spectrum disorder individuals undergoing 7 sessions of P300-based BCI joint-attention training, for a total of 105 sessions. The dataset was used for the 2019 IFMBE Scientific Challenge organized during MEDICON 2019 where, in two phases, teams from all over the world tried to achieve the best possible object-detection accuracy based on the P300 signals. This paper presents the characteristics of the dataset and the approaches followed by the 9 finalist teams during the competition. The winner obtained an average accuracy of 92.3% with a convolutional neural network based on EEGNet. The dataset is now publicly released and stands as a benchmark for future P300-based BCI algorithms based on multiple session data.

1.1. INTRODUCTION

A Brain-Computer Interface (BCI) is a system that provides a direct communication between the brain and a computer or external device [1]. In short, it must interpret brain activity and translate it into commands that can be used to control devices or programs, from prosthesis, orthosis, wheelchairs and other robots to a mouse or a keyboard in a controlled computer environment [2–4]. Different types of neuroimaging techniques can be used to implement BCIs, i.e., electroencephalography (EEG), magnetoencephalography (MEG), functional Magnetic Resonance Imaging (fMRI), functional Near-Infrared Spectroscopy (fNIRS), among others [5]. The most common modality is the EEG, since it provides a portable, inexpensive, non-invasive solution to measure brain activity with high temporal resolution [6–8,5].

There are several approaches to generate brain signals that can be interpreted and transformed into commands by the BCIs, namely event-related potentials (the most prominent being the P300), steady-state visual evoked potentials (SSVEP) or event-related synchronization/desynchronization (ERS/D) through mental imagery. The P300 approach, first attempted by Farwell and Donchin in the 80s [9], uses an oddball paradigm where an infrequent stimulus of interest is presented in a sequence of frequent stimuli of non-interest. With this paradigm, a positive deflection of the EEG measured in the central and posterior parts of the scalp is observed approximately around 300 ms after the infrequent stimulus of interest is presented [10,11]. The most common application of P300-based BCIs is the speller, where a matrix of letters flashing at different times is presented to the user. An infrequent event occurs due to selective attention to a specific target letter. Thus, a P300 potential is elicited whenever the letter the user is paying attention to flashes, and so the target letter can be identified by a P300 detection algorithm and then transmitted. The use-cases of P300-based BCIs have greatly increased over the past years, from steering a wheelchair [12] to composing music [13].

Despite the wide range of applications, there are still many challenges facing P300-based BCIs to be used more broadly. Achieving portable and practical BCIs that are easy to setup and fast to calibrate is currently a research line of big interest, since it would favorably help the adoption of this new technology in everyday settings [14,15,5]. However, different issues causing low robustness and reliability should be addressed for these systems to be used in real life. Indeed, often low performance is obtained by BCI models, even in laboratory conditions. The noise sensitivity, non-linearity and non-stationarity characteristics of EEG signals represent critical challenges since these properties depend both on the subject and the environment [16]. As a consequence of non-stationarity, shifts in EEG signals across trials and sessions occur. Therefore, robust feature extraction techniques are needed to overcome these perturbations on the signals [17]. Moreover, inter-subject variability, due to anatomical and physiological differences among subjects, also represents an important challenge since it hinders the design of participant-agnostic BCIs. Due to these main challenges (intra- and inter-subject variabilities), most BCIs require time-consuming calibrations to maximize their performance, which makes the creation of one-model-fits-all solutions difficult [18].

Nevertheless, the methods used for correctly identifying P300 signals have improved in the last years [19]. Traditional decoding algorithms rely on separate feature extraction and classification steps. Commonly used P300 features are based on temporal, time-frequency and spatial domains [20–22], while Linear Discriminant Analysis (LDA), Support Vector Machine

(SVM) and Multi-Layer Perceptron (MLP) are the most prominent classifiers used in P300-based BCI approaches. Some examples of recent improvements over traditional methods are the use of Riemannian geometry [23] or weightless neural networks [24]. Recently, deep learning techniques were transposed from the computer vision [25] to the EEG decoding field. Among these new solutions, Convolutional Neural Networks (CNN) and CNNs including recurrent layers - such as Long Short-Term Memories (LSTM) - on top of the convolutional extractor were used (CNN-LSTM) [26]. A key property of these algorithms is that they automatically learn the relevant features for a given task (i.e., the features are learned from the input data without any a priori feature extraction and selection) and finalize the target decoding task in an end-to-end fashion (i.e., without separating these steps). Nevertheless, these approaches pose some challenges: they require many hyper-parameters to be tuned (e.g., number of layers, number of kernels, etc.), they introduce a large number of parameters to be optimized during training (which are also difficult to interpret once trained) and thus, require the use of large datasets to achieve state-of-the-art decoding performance [26–28]. However, few datasets can be found in the literature matching this last requirement.

To evaluate the efficacy of new methods, authors need to compare their results with current state-of-the-art approaches. One viable approach is to implement both their method and established reference methods and apply all of them to the data of interest. Another option is to use benchmark datasets. Benchmark datasets are publicly available data usually launched in competition events where teams have the same information to start with and try to achieve the best possible result with their methods [29]. These competitions tend to disclose these datasets afterward, allowing both teams and other researchers to continue developing their methods and publish results that are comparable between them, if researchers recreate the original competition conditions on their attempts. Thus, these datasets provide a common ground for the research areas to assess their methods and improve the state-of-the-art.

One important contributor in this field has been the Berlin Brain-Computer Interface (BBCI) group through the organization of BCI competitions [30–33]. The corresponding datasets have been extensively explored and helped significantly the improvement of methods throughout the years [19,34]. Nevertheless, those datasets were limited in terms of subjects and sessions-per-subject, thus constraining the development of methods highly dependent on multi-session data.

In the scope of the XV Mediterranean Conference in 2019, the International Federation of Medical and Biological Engineering (IFMBE) launched a scientific competition based on a multi-session dataset of P300-based BCI intervention for young adults with autism spectrum disorder (ASD) [35]. This intervention was aimed at the rehabilitation of joint-attention, a core developmental skill that is altered in ASD and impacts other skills like language development [36]. Joint-attention refers to the ability of following social attentional cues of other people, so one's attention can be directed by the interlocutor to an external object or event of interest. Amaral et al. [14] developed an interventional BCI based on P300 signals that uses a virtual environment with a virtual human character and several objects of interest to train the ability of participants to follow the cues of the virtual character to the objects. That system was validated in an interventional pilot study [35] where 15 ASD individuals underwent 7 training sessions with this system. The database resulting from that interventional study supported the 2019 IFMBE scientific challenge and is now made public to the scientific community at https://

www.kaggle.com/disbeat/bciaut-p300 (doi: 10.34740/kaggle/dsv/1375326). This paper describes the challenge and corresponding dataset, summarizes the approaches by the competing teams and draws some conclusions from them, challenging the BCI research community to improve the current best performances achieved by the participating teams.

1.2. MATERIALS AND METHODS

1.2.1. Experiment Description

Overview of the P300-Based BCI System

The BCI system is composed mainly by two modules: data acquisition module and stimuli presentation module. For the data acquisition module, we used the g.Nautilus system (g.tec medical engineering GmbH, Austria) to record EEG data from 8 active electrodes positioned at C3, Cz, C4, CPz, P3, Pz, P4, POz locations. The reference electrode was placed at the right ear and the ground electrode at AFz location. Sampling rate was set to 250 Hz and data were acquired notch-filtered at 50 Hz and passband-filtered between 2 and 30 Hz. As for the stimuli presentation module, we used the Vizard toolkit to create and display a virtual environment consisting of a bedroom with common type of furniture (shelves, a bed, a table, a chair, and a dresser) and objects (frames, books, lights, a printer, a radio, a ball, a door, a window, and a laptop), as shown in Figure 1.1.

Figure 1.1 – Snapshot of the virtual environment, showing the scenario, the virtual avatar and the objects for joint-attention targets.

The objects used as stimuli throughout the experiment (and their respective labels) were: 1. books on a shelf, 2. a radio on top of a dresser, 3. a printer on a shelf, 4. a laptop on a table, 5. a ball on the ground, 6. a corkboard on the wall, 7. a wooden plane hanging from the ceiling, and 8. a picture on the wall. The virtual environment was presented via the Oculus Rift Development Kit 2 headset (from Oculus VR).

Each block consists of the user trying to identify one of the objects as the target. For that, K runs are repeated. One run is composed by a single flash of each object once for 100 ms at different times and random order, with an Inter-Stimulus Interval (ISI) of 200 ms. Figure 1.2 provides a schematic for this structure.

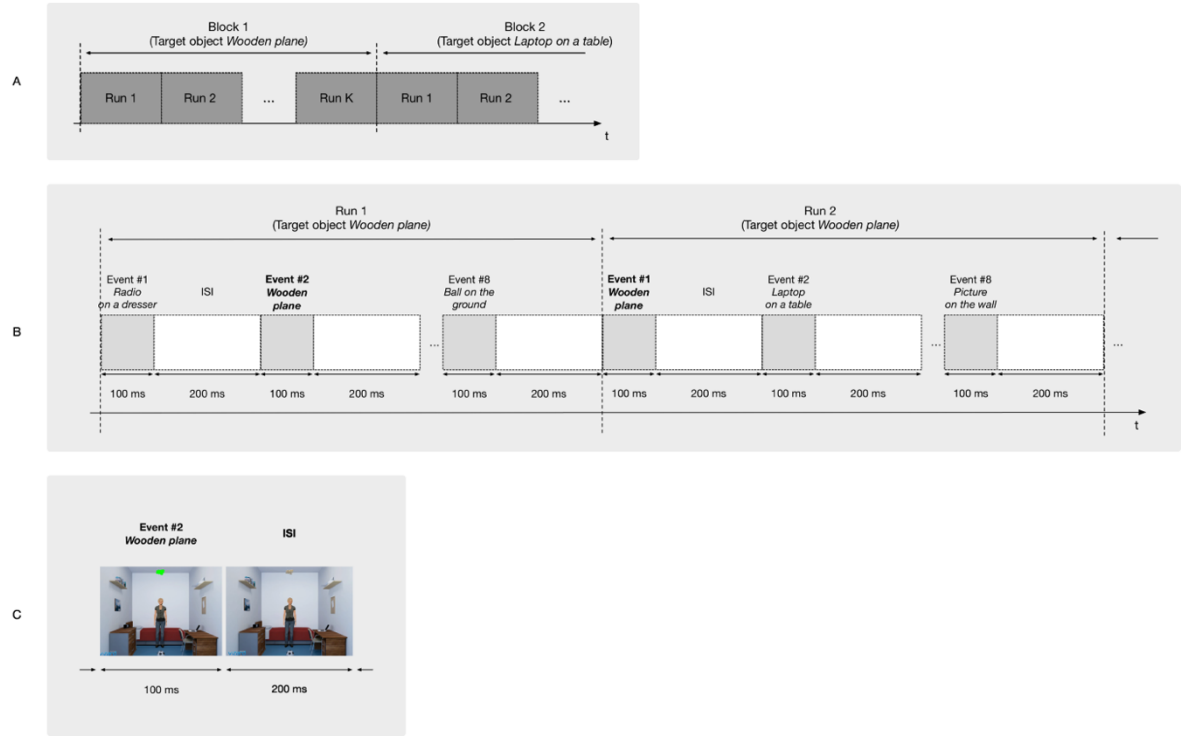


Figure 1.2 – Structure of the paradigm with its subdivisions in blocks, runs and events. (A) Structure of the blocks: each block is used to identify a single target object and is composed by K runs. (B) Structure of the runs: each run is composed by 8 events, each consisting of the flashing of one of the objects. (C) Structure of an event: it consists of the flashing of the corresponding object by 100 ms, followed by an interval of 200 ms.

BCI Session Flow

Fifteen participants performed 7 identical training sessions in different days, the first four on a weekly basis and the last three on a monthly basis. Each training session was divided in two parts: calibration and online phase. Data from calibration and online phases were named in the dataset as train and test data, respectively.

The calibration phase was composed of 20 blocks, each block containing 10 runs. Because we used 10 runs per block, a total of 200 target P300 signals and 1400 non-target signals were acquired at this phase. With these data, the session-specific classifiers were trained for the online phase and the number of runs per block (K) to use on the online phase was defined. K was defined during the online sessions of the clinical trial as the minimum number of runs for which the classifier achieved an accuracy above 80%, in the calibration data.

Regarding the online phase, 50 blocks were taken for each participant using K runs per block. The value of K varied between subjects and sessions, since it was an output of the calibration phase, ranging from 3 to 10.

1.2.2. Dataset Structure and Contents

The dataset folder structure is organized by subjects, with a folder for each subject named SBJXX, with XX varying from 01 to 15. Within each subject folder there is a set of folders containing the data from each session, named SYY, with YY varying from 01 to 07. Each session folder contains a separate folder for the training and testing data, named Train and Test, respectively.

i. Train folder
   - trainData.mat – Data from the calibration phase, structured as [channels x epoch x event], epoch corresponding to the data samples from −200 ms to 1000 ms relative to the event stimulus onset (epoch length of 1200 ms; 300 data samples).
   - trainEvents.txt – One label per line (from 1 to 8), corresponding to the order of the flashed objects.

ii. Test folder
   - testData.mat – Data from the online phase, in the same structure as the train data.
   - testEvents.txt – One label per line (from 1 to 8), corresponding to the order of the flashed objects.
   - testTargets.txt – 1 or 0 per line, indicating if the flashed object was the target or not, respectively.
   - testLabels.txt–Label of the target object per line(from 1 to 8), one for each block.
   - runs_per_block.txt – File containing only one number, corresponding to the number of runs per block used in the online phase (from 3 to 10).

The number of epochs corresponds to # events per run * # runs per block * # blocks. For the training data, it represents 8 events per run * 10 runs per block * 20 blocks = 1600 epochs. As for the test data, since the number of runs varies between sessions, the number of epochs varies in consequence, in a total of 8 events per run * K runs per block * 50 blocks = 400 * K epochs.The channels' order in the data matrices is C3, Cz, C4, CPz, P3, Pz, P4, POz. The first sample of each epoch corresponds to the time −200 ms relative to the stimulus onset and the last sample to corresponds to the time 996 ms after the stimulus onset (the last sample < 1000 ms), with a sampling rate of 250 Hz, for a total of 300 samples.

1.2.3. Challenge Structure

For the 2019 IFMBE Scientific Challenge, teams were asked to maximize the P300-based object detection accuracy for the 7 sessions of the 15 ASD participants of the BCIAUT clinical trial. For each session, a train and test set were created, without disclosing the true labels of the test sets. The challenge was divided into two phases with a different number of attempts per phase (Table 1.1). For phase I, sessions 1–3 were provided, without the test labels. At the end of phase I, the true test labels of those three sessions were made available to the participants along with the remaining 4 sessions (4–7), the latter without the true test labels (phase II). This way, teams could use the true labels of the first three sessions to improve their classifiers, if working with multi-session data. Teams were allowed to submit 5 attempts during phase I and 10 attempts during phase II. The best submission of each team throughout the allowed attempts on each phase was used to rank the teams. The complete dataset (including all true labels) is now available at https://www.kaggle. com/disbeat/bciaut-p300 (doi: 10.34740/kaggle/dsv/1375326).

Table 1.1 – Timetable and number of attempts for the two phases of the competition.

Phase	Start Date	End Date	Number of Attempts
Phase I	01-03-2019 10:00	15-05-2019 23:59	5
Phase II	20-05-2019 10:00	30-06-2019 23:59	10

1.2.4. Submissions and Approaches

Fourteen teams participated in phase I of the competition, while 9 teams participated in phase II and concluded the challenge. The results shown in this manuscript refer to the phase II of the competition. The performance metric used to compare the performance of contesting teams was the target object detection accuracy, computed as the ratio between the number of correct predicted blocks and the total number of blocks to decode. Based on the average target object accuracy across subjects and sessions, the approaches proposed by each team were ranked up.

The following list of IDs reflects the final ranking of the competition:

- ID-1: D. Borra, S. Fantozzi and E. Magosso [37]
- ID-2: E. Santamaría-Vázquez, V. Martínez-Cagigal, J. Gomez-Pilar and R. Hornero [38].
- ID-3: L. de Arancibia, P. Sánchez-González, E. J. Gómez, M. E. Hernando and I. Oropesa [39].
- ID-4: M. Bittencourt-Villalpando and N. M. Maurits [40].
- ID-5: D. Krzemiński, S. Michelmann, M. Treder and L. Santamaria [41].
- ID-6: A. Miladinovic´, M. Ajc´evic´, G. Silveri, G. Ciacchi, G. Morra, J. Jarmolowska, P. P. Battaglini and A. Accardo [42].
- ID-7: B. Chatterjee, R. Palaniappan and C. N. Gupta [43].
- ID-8: V. S. Adama, S. Benjamin and T. Schmid [44].
- ID-9: H. Zhao, S. Yu, J. Prinable, A. McEwan and P. Karlsson [45].

For each team, a brief description of the proposed methodology is reported:

- ID-1: Epochs were extracted between −100–1000 ms, and the signals were downsampled to 128 Hz. The decoding solution was based on a CNN performing classification at the level of single trial (EEG response to a single stimulus, without averaging). The input was a 2-D representation composed by the EEG channels along one dimension (spatial dimension) and time steps along the other dimension (temporal dimension). The CNN was an adaptation of EEGNet [27] trained to discriminate between P300 and non-P300 classes. In this CNN design, depthwise and pointwise convolutions are used to keep the number of trainable parameters limited. The architecture in its fundamental subnetworks and main connections between neurons is displayed in Figure 1.3. Furthermore, a detailed description of these subnetworks including the main hyper-parameters, output activation shapes and number of trainable parameters introduced is reported in Table 1.2. The CNN is composed by 3 main subnetworks (here labeled as A, B, C), performing different operations on the input. These include a temporal and spatial feature extractor (Figure 1.3A) that learns meaningful temporal and spatial filters, a summary feature extractor (Figure 1.3B) that learns to extract temporal summaries for each feature map of the subnetwork A individually; and a classification module (Figure 1.3C) that finalizes the classification task based on the output of the subnetwork B. The obtained single-trial probabilities were then averaged together across runs related to a specific object

belonging to each block, and then the object with maximum average probability was selected, solving the target 8-way classification task. Different intra-subject training strategies were explored, including inter-session (i.e., training subject-specific classifiers) and intra-session (i.e., training session-specific classifiers) training strategies. The top-performing solution of ID-1 was the one adopting a subject-wise inter-session strategy. The code of the CNN and the weights of the trained models are available at https://github.com/ddavidebb/ IFMBE2019Challenge-BCIAUT-P300.

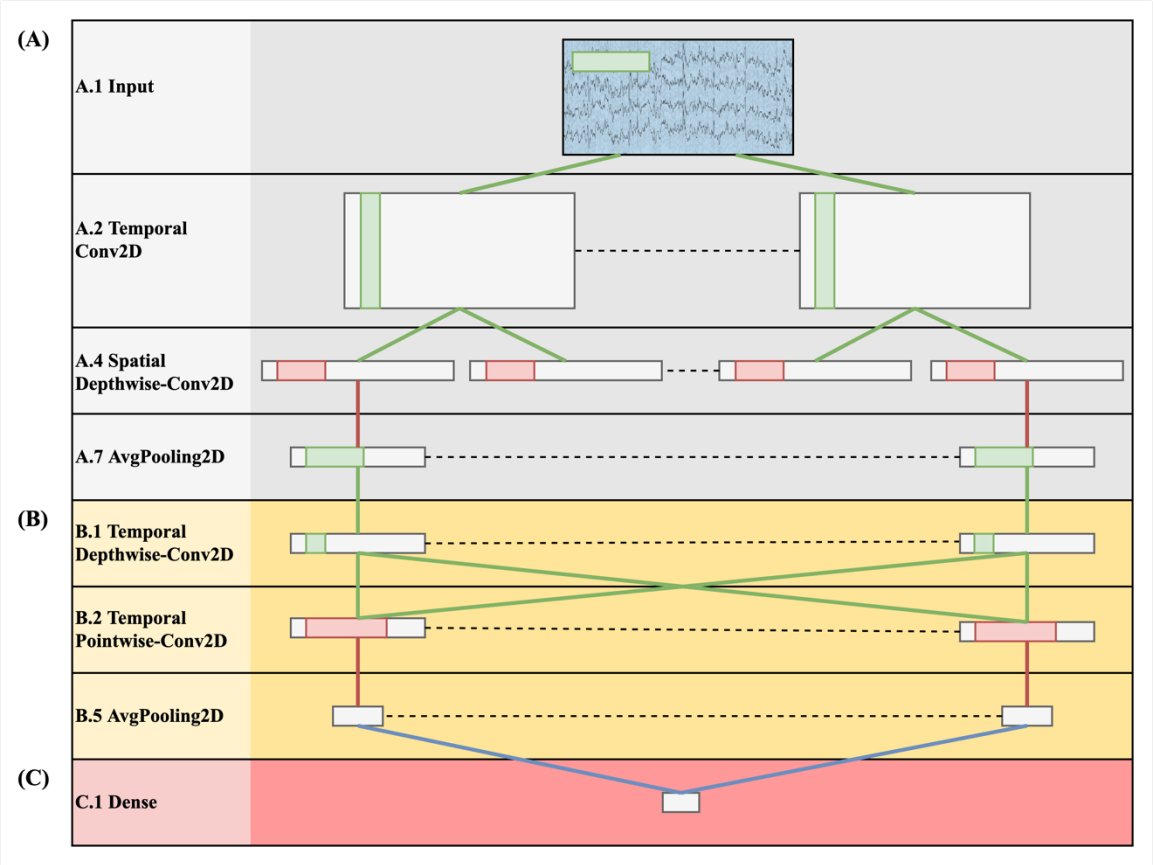


Figure 1.3 – Architecture schematization of the winning solution ID-1 based on EEGNet. The represented shapes correspond to the output of each layer. Green lines represent convolutional connections, red lines pooling connections, and blue lines dense connections. The CNN is composed by a temporal and spatial feature extractor (A), a summary feature extractor (B) and a classification module (C).

Table 1.2 – Architecture design inspired from EEGNet and adopted in ID-1. K and F are the number and the size of the kernels, respectively. P is the padding size, D the depth multiplier, N the number of neurons in the dense layer and finally p the dropout rate. Light gray denote layers with trainable parameters. The total number of trainable parameters is 1386. *Unitary kernel max-norm constraint.

Subnet.	Layer ID	Layer	Hyper-parameters	# pars	Output shape	Activation
A	A.1	Input		0	(1,8,140)	
	A.2	Temporal Conv2D	K=8, F=(1,65), P=(0,32)	520	(8,8,140)	Linear
	A.3	BatchNorm2D		16	(8,8,140)	
	A.4	Spatial Depthwise-Conv2D*	D=2, K=16, F=(8,1), P=(0,0)	128	(16,1,140)	Linear
	A.5	BatchNorm2D		32	(16,1,140)	

	A.6	Activation		0	(16,1,140)	Exponential Linear Units (ELU)
	A.7	AvgPooling2D	F=(1,4)	0	(16,1,35)	
	A.8	Dropout	p=0.25	0	(16,1,35)	
B	B.1	Temporal Depthwise-Conv2D	D=1, K=16, F=(1,17), P=(0,8)	272	(16,1,35)	Linear
	B.2	Temporal Pointwise-Conv2D	K=16, F=(1,1), P=(0,0)	256	(16,1,35)	Linear
	B.3	BatchNorm2D		32	(16,1,35)	
	B.4	Activation		0	(16,1,35)	ELU
	B.5	AvgPooling2D		0	(16,1,4)	
	B.6	Dropout	p=0.25	0	(16,1,4)	
C	C.1	Dense	N=2	130	(2,)	Linear
	C.2	Activation		0	(2,)	Softmax

- ID-2: EEG signals were epoched between 0–1000 ms, applying a baseline (−200-0 ms) normalization. The input representation is the same as in ID-1. The task was aced as a 2-way classification decoding P300 and non-P300 classes for each trial adopting an adaptation of the CNN proposed by Manor et al. [46], a CNN-LSTM and a CNN-BLSTM. Furthermore, these deep learning architectures were compared with a more traditional machine learning pipeline including SWLDA. The top-performing algorithm proposed by ID-2 was CNN-BLSTM. This network was composed of one convolutional layer 1-D that extracts spatio-temporal patterns on the input, two bidirectional LSTM layers and one dense layer. The single-trial probabilities were averaged to obtain object-level probabilities as in ID-1. An intra-subject and inter-session training strategy was adopted, training subject-specific classifiers. The code of the models and the weights of the trained models are available at https://github.com/esantamariavazquez/IFMBE2019Challenge-BCIAUT-P300.
- ID-3: EEG signals related to a specific object were averaged across trials of the same block. Feature extraction was based on temporal and time-frequency parameters. Temporal features were extracted in epochs between 0–1000 ms by downsampling the signals with a decimation factor of 10. In addition to temporal features, features based on continuous wavelet transform (CWT) were extracted from epochs between 200–712 ms. The t-CWT was computed based on a Mexican Hat wavelet on scales corresponding to the delta (0.5–4 Hz) and theta (4–8 Hz) bands [20,21]. These temporal and time-scale features were concatenated across channels in a single vector. Principal component analysis (PCA) was applied for feature dimensionality reduction, which resulted in a final vector of 120 features. A comparison of different combinations of linear and non-linear machine learning approaches was performed. More specifically, linear discriminant analysis (LDA) and support vector machines with linear kernel (LSVM), and a more complex support vector machine with radial kernel (RSVM) were employed. The object whose corresponding signals yielded a higher probability of containing a P300 event was chosen as predicted target object of the block. In addition, the effect on the accuracy of the number of EEG events averaged was studied. An inter-session training strategy was adopted, comparing both subject-specific and inter-subject classifiers, as well as the use

of oversampling and boosting techniques to account for class imbalance. LDA outperformed the other classifiers and was used to classify the target object. Best results were obtained for > 3 events averaged. Training subject-specific classifiers yielded the best performance. Oversampling and boosting did not improve the final performance of the classifiers. The developed code and trained models are available at: dev.gbt.tfo.upm.es/ioropesa/ifmbe-scientific-challenge-competition---detection-of-p300/tree/master.

- ID-4: The approach consisted of the adaptation and parameter optimization of an SVM-based algorithm that was previously developed for a 4-choice BCI [47] for target identification. During the first phase of the challenge, the original algorithm was adapted for 8 choices and the pre-processing parameters were defined as follows. First, temporal features were extracted in epochs between 0–1000 ms following each event onset and all channels were concatenated in a single feature vector per event for each participant and session. Then, feature vectors containing EEG signals from target events were pseudo-randomly averaged across blocks belonging to the same session for noise reduction. During the second phase of the challenge, an intra-subject and intra-session training strategy was developed, augmenting the dataset with other sessions' signals, and artificially increasing the number of targets per session by adapting the pseudorandom averaging procedure. Eight parameters related to data augmentation and SVM input parameters were optimized throughout the 9 initial attempts and then compared in terms of accuracy. The parameters' description and settings per attempt are detailed in [40]. In the last attempt, the best performing parameter setting was selected, resulting in a customized solution per participant and per session.
- ID-5: This solution exploited Riemannian framework for EEG signal decoding [23]. The approach was computationally efficient and recently outperformed other common state-of-the-art approaches [48]. The Riemannian framework was combined with the ensemble learning. The idea was to build upon many "weak" (under-performing) classifiers and then combine their outcomes to improve the performance of the final model. The ensemble of 8 different data features was constructed by combining 2 different band-pass filters (1– 20 Hz or 1–8 Hz), 2 trial lengths (from −200 to 1000 ms or from 0 to 600 ms) and 3 different subsets of electrodes (all, central or posterior only electrodes). Then, the ERP prototypes were created by calculating the ERP for each channel. Next, the regularized covariance matrices of a single trial concatenated with the prototype were computed and the resultant matrices were projected into the tangent space of a reference matrix. Fisher Geodesic Discriminant Analysis (FGDA) was used to project the matrices to a lower-dimensional discriminative subspace. The resultant projections were flattened to vectors and used as the features to the ensemble learning algorithm comprising 400 LDA classifiers. The output probability was aggregated across trials belonging to each object to decode the target per each block. An intra-subject and intra-session training strategy was adopted. The developed code is available at https://github.com/dokato/bci-challange.
- ID-6: The windows mean approach was used to obtain the temporal features on each trial. These were computed for each electrode on 50 ms windows without overlap from 100–1000 ms. Bayesian logistic regression with automatic relevance determination (VB-

ARD) [49] was used to classify the P300 event on each trial. The method has an advantage over other regularization techniques which need a separate validation set to eliminate irrelevant features. Besides, this approach generates a posterior distribution enabling the authors to model the varying-intercept sparse feature model. The modeling applied in this approach is similar to the one proposed by [50] with a variation of Automatic Relevance Determination (ARD) that instead of using type-II maximum likelihood [51], applies full Bayesian treatment [49]. The primary generative model matches the one employed in [50], and the prior is selected to be non-informative, modeled by a conjugate Gamma distribution [49]. This makes the model parameter-free and easy to use without deep knowledge in the data science domain. The advantage of this methodology is that obtained distribution allows the authors to find the inverse of the predictors' covariance matrix (precision matrix) and apply Automatic Relevance Determination (ARD) that assigns an individual hyper-prior to each regression coefficient separately determining their relevance and produces for each trial a class-belonging probability. Lastly, single-trial probabilities were averaged together across trials for each object belonging and the one with maximum average probability was selected. In this method, an intra-subject and intra-session training strategy was performed. The demo code is available at https://github.com/miladinovic/ BCILabTS under subfolder userscripts.

- ID-7: Whole signals were used (−200–1000 ms) and the pre-stimulus mean (−200-0 ms) was removed. Signals were filtered between 2–12 Hz and the filtered signals were downsampled 10-times. Then, these downsampled electrode signals were normalized epoch-wise in the range −100–1000 ms. These temporal features were used to classify the P300 event for each trial with BLDA, RUSBoost and CNN. The best performing classifier for each subject was used (subject-specific classifier). Then, a majority voting was done to determine the target object within each specific block. An intra-subject and inter-session training strategy was performed.
- ID-8: EEG signals were averaged across trials related to a specific object belonging to each block. Temporal features were extracted for each electrode by averaging for each time window from 200–450 ms and decimating the output with a factor of 12. In addition, Pearson's correlation coefficient was computed for each electrode between the time window of interest and the time window preceding stimulus presentation (−200-0 ms). These temporal features and correlation coefficients were concatenated across channels in a single feature vector. An inter-subject and inter-session training strategy was performed, by which a variety of competing supervised learning techniques (decision tree, random forest, SVM, MLP) were trained to classify the target object within each block. From those, MLP performed best on the given data.
- ID-9: Epochs were extracted from 0–600 ms. An additional 20 Hz low-pass filter was applied to the original data. In addition, a custom filter was designed to address each subject- and session-specific noise features. The temporal features were selected using a linear support vector regression as a pre-selector for features in the data. A comparison between linear and non-linear methods was performed, using SVM, LDA, 1D 4-layer CNN, 1-layer LSTM. LDA was the top-performing classification algorithm for ID-9 and was used to classify the P300 event for each trial. Then, the label that appeared most

times within each block was the target object to decode. An intra-subject and intra-session training strategy was adopted. The code is available at https://github.com/hyphenzhao/MEDICON2019ScientificChallenge.

A summary of the top-performing method of each team adopted for the challenge is shown in Table 1.3.

1.2.5. Statistical Analysis

For each team, the best-performing solution proposed among the phase II attempts – in terms of target object accuracy averaged across subjects and sessions – was selected for analysis and the algorithms were then ranked up based on this average score. Furthermore, the metrics scored by algorithms ID-2:9 were compared with the winning algorithm (ID-1) using Wilcoxon signed-rank tests. To correct for multiple tests, a false discovery rate correction at 5% using the Benjamini-Hochberg procedure [52] was applied and the corrected p-values are reported.

Table 1.3 – Summary of the best-performing algorithm of each team developed for the challenge.

ID #	acc (%)	Pre-processing	Methodology	Post-processing	Training strategy	Framework
ID-1	92.3±1.8	• Epochs from -100 to 1000 ms • Downsampling to128 Hz	• CNN based on EEGNet [27]	• Average probability across runs within a specific block • Decoding of the target object as the object with maximum average probability	• Intra-subject and inter-session	• Python with PyTorch
ID-2	84.3±3.2	• Epochs from 0 to 1000 ms • Baseline normalization from -200 to 0 ms	• CNN-BLSTM	• Average probability across runs within a specific block • Decoding of the target object as the object with maximum average probability	• Intra-subject and inter-session	• Python with Scikit-learn and Keras

ID-3	82.0±2.5	• Temporal features:
		  ○ Ensemble averaging per block
		  ○ Temporal epoching from 0 to 1000ms
		  ○ Moving-average downsampling
		• CWT features:
		  ○ Temporal epoching from 200ms to 712ms
		  ○ Most differential points computed with t-Student (t-CWT)
			• Temporal features concat (200 features)
			• Computation of the t-CWT [21] based on Mexican Hat wavelet (128 points per channel) and CWT features concat. (1024 features)
			• Feature reduction based on PCA (120 features)
			• LDA
				• The object whose corresponding signals yield a higher probability of containing a P300 was chosen as predicted target object of the block	• Intra-subject and inter-session	• MATLAB with Statistics and Machine Learning Toolbox and Signal Processing Toolbox

ID-4	81.5±2.6	• Epochs from 0 to 1000 ms • Pseudorandom • averaging of ERP segments.	• Feature vector with 2000 elements per ERP (concat. of 8 channels*250 elements) • SVM	• The feature vectors were sorted according to the event (flashed object, from 1 to 8) • All runs per block were averaged, per event • The predicted target corresponds to the event with the highest score.	• Intra-subject and intra-session • Data augmentation with other sessions' signals and with pseudorandom averaging	• MATLAB with Statistics and Machine Learning Toolbox 2017.

ID-5	81.2±2.1	• Band-pass filtering with two different filters (1-20 Hz or 1-8 Hz) and two variations of trial length (whole signal or the first 600 ms after stimuli onset) • Three subsets of electrodes were chosen (all, central or posterior electrodes)	• ERP prototypes were created by calculating the ERP for each channel • Regularized covariance matrices of a single trial signal concatenated with prototype were calculated • The resultant covariance matrices were projected into the tangent space of a reference matrix • FGDA was used to project the matrices in tangent space to a lower-dimensional discriminative subspace. These were used as features. • Ensemble of 400 LDA classifiers (taking 40% of data samples and 60% of features) operated on ensemble of signal preprocessed in 8 different combinations	• Aggregated probability of trial belonging to each of the classes.	• Intra-subject and intra-session	• MATLAB
ID-6	80.3±2.2	• Epochs from 100 to 1000 ms	• Temporal features computed on 50 ms windows, without overlap, producing 18 features per channel for each event • VB-ARD	• Average probability across runs within a specific block • Decoding of the target object as the object with maximum average probability	• Intra-subject and intra-session	• MATLAB • BCILAB

ID-7	76.3±2.9	• Epochs from -200-1000 ms
		• Pre-stimulus mean (-200-0 ms) was removed.
		• Band-pass filtering 2-12 Hz
		• Normalization epochwise to the interval [-1,1]
			• Temporal features were extracted by downsampling with a factor of 10 the normalized and filtered signals
			• Three classifiers were trained and tested:
			  ○ BLDA
			  ○ RUSBoost
			  ○ CNN
				• The best performing classifier for each subject was used
				• Majority voting within each run to determine which flash has been classified as target maximum number of time and that was predicted as target for that particular run
					• Intra-subject and inter-session	• MATLAB with Classification App RUSBoosted Trees

ID-8	70.0±3.8	• Averaging of EEG signals across trials related to a specific object within each block	• Temporal features (based on [53]): averaging within windows from 200-450ms; 56 features per channel (448 total) • Pearson's correlation coefficients: coefficients were computed between the time window of interest and the time window preceding stimulus presentation (-200-0 ms); 8 features per channel (64 total) • Concatenation of temporal and pearson's coefficients across channels in a single feature vector MLP	-	• Inter-subject and inter-session	• MATLAB (pre-processing) • Python with Scikit-learn (main algorithm)
ID-9	67.2±3.3	• Epochs from 0-600 ms • Low-pass filter 20 Hz • Custom filter to address each subject- and session-specific noise features deduced from non-target epochs	• Linear support vector regression as feature pre-selector • LDA	• The label that appeared most times within each block was the target object to decode	• Intra-subject and intra-session	• Python with Scikit-learn

1.3. RESULTS

In Tables 1.4, 1.5 the accuracies of the proposed approaches are shown, describing the decoding variability across subjects and recording sessions. In particular, Table 1.4 reports for each subject the average target object accuracy across sessions (i.e., performance at the level of single subjects), while Table 1.5 reports for each session the average target object accuracy across subjects (i.e., performance at the level of single session).

Table 1.4 – Performance at the level of single subject as represented by the average target object accuracies of the best approach proposed by each team. The mean accuracy (acc) and its standard error (SEM) are reported. Wilcoxon signed-rank test was used to compare ID-1 with ID-2:9 and the corrected p-values for multiple tests are reported (*p<0.05, **p<0.01, ***p<0.0001).

ID #	Accuracy at the level of single subject (%)															
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**acc (mean±SEM)**
ID-1	81	100	86	96	93.5	96	96.5	100	90.5	98	94	84.5	86.5	81.5	100	92.3±1.8
ID-2	56	98	67.5	96	80	88	86.5	99	82	93	87.5	80	81	71.5	98.5	84.3±3.2**
ID-3	73	95	71	91	82.5	86	85	91.5	68.5	88.5	86	80.5	60	84	87.5	82.0±2.5***
ID-4	64.5	92	68	94.5	84	86	81.5	94	71	87	87	82	66	77	88	81.5±2.6***
ID-5	69	91	67	88.5	79.5	82.5	83	95	82.5	81.5	85.5	79	69	78	87.5	81.2±2.1***
ID-6	68	91.5	71.5	92.5	80	84	79	94.5	73.5	82.5	84	78.5	68.5	71.5	85.5	80.3±2.2***
ID-7	54	93	62.5	90	73	85.5	76	88.5	71	78	80.5	78	65	65	84.5	76.3±2.9***
ID-8	48	84	58	69	69.5	52	84	94	72	87.5	77	64	50	56.5	84.5	70.0±3.8***
ID-9	46	85	53	77	65	66.5	67.5	89	57	72	73.5	73	59.5	47	77.5	67.2±3.3***

Table 1.5 – Performance at the level of single session as represented by average target object accuracies across subjects of the best approach proposed by each team. The mean accuracy (acc) and its standard error (SEM) are reported. Wilcoxon signed-rank test was used to compare ID-1 with ID-2:ID-9 and the corrected p-values for multiple tests are reported.

ID #	Accuracy at the level of single session (%)							
	4		**5**		**6**		**7**	
	acc (mean±SEM)	**p-value**	**acc (mean±SEM)**	**p-value**	**acc (mean±SEM)**	**p-value**	**acc (mean±SEM)**	**p-value**
ID-1	92.8±2.4	-	90.4±3.5	-	94.8±1.8	-	91.1±3.0	-
ID-2	85.1±3.1	0.0044	82.0±5.5	0.0026	90.5±2.6	0.0082	79.6±5.6	0.0026
ID-3	81.5±3.3	0.0023	82.0±4.4	0.0062	84.3±2.6	0.0025	80.3±3.7	0.0015
ID-4	80.3±3.0	0.0015	80.7±4.4	0.0037	84.9±2.6	0.0015	80.1±4.2	0.0020
ID-5	79.9±3.3	0.0013	78.4±4.2	0.0015	85.1±2.4	0.0013	81.6±4.2	0.0032
ID-6	78.1±3.6	0.0015	79.6±4.0	0.0017	83.6±2.6	0.0013	80.0±3.7	0.0013
ID-7	75.2±3.8	0.0013	72.8±4.9	0.0013	80.3±2.6	0.0013	76.9±3.6	0.0013
ID-8	70.5±4.3	0.0013	70.3±6.0	0.0013	72.7±3.8	0.0013	66.5±5.7	0.0013
ID-9	64.8±4.3	0.0013	66.9±4.5	0.0013	69.3±4.2	0.0013	67.9±5.4	0.0013

Averaging across sessions and across subjects, ID-1 significantly outperformed the other approaches, with less variability across subjects and sessions. Looking at the performance at the level of subjects, ID-1 provided the best performance metric for 14 out of 15 subjects (for subject #4, ID-2 provided a top-performance across the proposed solutions too), while ID-3 provided the best performance metric for 1 out of 15 subjects (subject #14). Averaging across subjects, ID-1 significantly outperformed the other approaches within each recording session, with less variability across subjects and providing an average performance above 90% for all the phase II sessions.

1.4. DISCUSSION

In this study, a large multi-session and multi-subject dataset acquired during a P300-based BCI intervention for young adults with ASD was presented. The evolution and the practical application of deep learning solutions for EEG decoding depend on the availability of large multi-subject datasets. Furthermore, the lack of multi-session datasets hinders the design of reliable algorithms across recording sessions. Thus, the described dataset represents a multi-session collection of signals that can be used as a benchmark to design accurate and reliable data-hungry algorithms, such as deep learning solutions, for P300 decoding tasks.

In fact, the richness of the dataset enabled the use of deep learning approaches in the context of the competition. Among the proposed algorithms, a deep learning solution based on a lightweight CNN (see ID-1 in Section *Submissions and Approaches*) outperformed both a CNN-BLSTM ($p = 0.001$, across subjects and sessions, see Table 1.4, ID-2) and more traditional machine-learning solutions ($p < 0.001$, across subjects and sessions, see Table 1.4). Furthermore, this was found also for single session recordings ($p < 0.005$ when comparing ID-1 with other solutions, see Table 1.5), with average metrics above 90% (far above the chance level of 12.5%). The best non-deep learning solution adopted temporal and CWT features, alongside with PCA for dimensionality reduction and LDA for classification (see ID-3 in Section *Submissions and Approaches*). The training strategies performed in the approaches ID-1:3 were both intra-subject and inter-session. In particular for the winning solution, from the experiments between inter-session and intra-session trainings performed by ID-1, better results were found using all the session signals during the optimization.

When using deep learning approaches with EEG signals, the input representation and the design of spatio-temporal convolutions is not trivial and need to be addressed. Regarding the input representation, the time series are related to electrodes placed on a 3D surface. Typically, EEG signals can be represented in three different ways to feed the input layer of a neural network [27]:

a. Using the original representation of all the available electrode signals to design a 2D representation where EEG channels are reported along one dimension (spatial dimension) and time steps along the other dimension (temporal dimension).
b. Using a transformed representation (e.g., time-frequency decomposition) of all the available electrodes.
c. Using a representation as in (b) with a subset of electrodes.

Among these representations, the first one is preferred since a representation like (b) generally increases the dimensionality [27], leading to more trainable parameters and, thus, to the need of more data or an increased regularization. Furthermore, several hyper-parameters are introduced depending on the transformation applied. Lastly, representations like (c) share the main disadvantages of (b) with an additional needing of a priori knowledge about the more relevant subset of electrodes to choose. Therefore, representations that respect the scheme (a) are a good compromise between input dimensionality and capability to learn more general EEG features on all the electrode signals [27]. Among the best-performing solutions in this competition, ID-1 and ID-2 adopted the first input representation scheme.

Regarding the design of spatio-temporal convolutions, depending on the information processing in the convolutional module, three different solutions can be designed starting from the input layer:

i. The temporal filtering is performed at first and then the spatial filtering.
ii. The spatial filtering is performed at first and then the temporal filtering.
iii. Mixed spatio-temporal filtering.

The CNN adopted by ID-1 used the convolutions ordering as in point i., while the CNN-BLSTM adopted by ID-2 as in (iii). Furthermore, among the solutions proposed by ID-2, there was a CNN based on [46] adopting a convolution ordering as in point ii.. Thus, in this competition, the solutions based on convolution ordering as in point i. outperformed the solutions following designs as in points i. and ii.

In addition, the layers of the neural network need to be carefully designed to keep control the number of trainable parameters and thus, to avoid overfitting when handling a limited collection of training signals. To this aim, architectures like EEGNet [27] were proposed including optimized convolutions, such as depthwise and separable convolutions [54]. The CNN adapted in ID-1 was inspired from [27] and introduced only 1386 trainable parameters, while the CNN-BLSTM designed by ID-2 introduced 10113 parameters. Lastly, among the solutions proposed by ID-2 (different from the best-performing algorithm of ID-2), a CNN based on Manor et al. [46] introduced 37428963 parameters. Therefore, in this competition the use of a lightweight architecture to solve the target P300 decoding task was beneficial. This result is in line with the recent growth of interest in the design of optimized layers in CNNs for EEG decoding as proposed by [55,56].

The BCIAUT-P300 dataset presents rare characteristics which reinforce its potentialities to work as a benchmark for P300-based BCI methods: 1) the multi-subject dimension, with 15 participants undergoing the same procedure, enable the possibility of developing inter-subject methods for generalized off-the-shelf applications; 2) the multi-session dimension, since each subject repeated the same training task 7 times in different weeks, enables the study of stability and reliability of subject-specific BCI methods throughout time, and even the inclusion of reinforcement learning strategies by approaching the sessions gradually; and 3) the ASD clinical dimension, since real-life BCI applications on ASD patients pose several challenges, this dataset provide a test bench for data quality and artifactual EEG data on ASD population that new projects can use to validate its models before approaching the clinical patients directly.

1.5. CONCLUSIONS

This paper presented the BCIAUT-P300 dataset which combines multi-session and multi-subject data of 15 ASD participants using a P300-based BCI for training joint-attention skills. The dataset was used on the IFMBE scientific competition where 9 teams from around the world reach the final phase and presented their methods, which were briefly presented here. Overall, deep learning methods were able to overcome the more traditional machine learning approaches, with the best method obtaining an average accuracy of 92.3%. Future studies should address the multiple dimensions of the dataset to reduce training times while improving accuracy.

1.6. REFERENCES

[1] Wolpaw J and Wolpaw E W 2012 *Brain-Computer Interfaces: Principles and Practice* (Oxford University Press, USA)

[2] Bamdad M, Zarshenas H and Auais M A 2015 Application of BCI systems in neurorehabilitation: a scoping review *Disability and Rehabilitation: Assistive Technology* **10** 355–64

[3] Chaudhary U, Birbaumer N and Ramos-Murguialday A 2016 Brain–computer interfaces for communication and rehabilitation *Nature Reviews Neurology* **12** 513–25

[4] McFarland D J and Wolpaw J R 2017 EEG-based brain–computer interfaces *Current Opinion in Biomedical Engineering* **4** 194–200

[5] Zou Y, Zhao X, Chu Y, Zhao Y, Xu W and Han J 2019 An inter-subject model to reduce the calibration time for motion imagination-based brain-computer interface *Med Biol Eng Comput* **57** 939–52

[6] Sitaram R, Caria A, Veit R, Gaber T, Rota G, Kuebler A and Birbaumer N 2007 fMRI Brain-Computer Interface: A Tool for Neuroscientific Research and Treatment *Computational Intelligence and Neuroscience* **2007** 1–10

[7] Bhattacharyya S, Khasnobish A, Ghosh P, Mazumder A and Tibarewala D 2017 A review on brain imaging techniques for BCI applications *Medical Imaging: Concepts, Methodologies, Tools, and Applications* (IGI Global) pp 300–30

[8] Deshpande G, Rangaprakash D, Oeding L, Cichocki A and Hu X P 2017 A New Generation of Brain-Computer Interfaces Driven by Discovery of Latent EEG-fMRI Linkages Using Tensor Decomposition *Front. Neurosci.* **11** 246

[9] Farwell L A and Donchin E 1988 Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials *Electroencephalography and Clinical Neurophysiology* **70** 510–23

[10] Riggins T and Scott L S 2020 P300 development from infancy to adolescence *Psychophysiology* **57**

[11] Guo M, Jin J, Jiao Y, Wang X and Cichockia A 2019 Investigation of Visual Stimulus With Various Colors and the Layout for the Oddball Paradigm in Evoked Related Potential-Based Brain–Computer Interface *Front. Comput. Neurosci.* **13** 24

[12] Lopes A, Rodrigues J, Perdigao J, Pires G and Nunes U 2016 A New Hybrid Motion Planner: Applied in a Brain-Actuated Robotic Wheelchair *IEEE Robot. Automat. Mag.* **23** 82–93

[13] Pinegger A, Hiebel H, Wriessnegger S C and Müller-Putz G R 2017 Composing only by thought: Novel application of the P300 brain-computer interface ed P Sardo *PLoS ONE* **12** e0181584

[14] Amaral C P, Simões M A, Mouga S, Andrade J and Castelo-Branco M 2017 A novel Brain Computer Interface for classification of social joint attention in autism and comparison of 3 experimental setups: A feasibility study *Journal of Neuroscience Methods* **290** 105–15

[15] Nakanishi M, Wang Y-T, Wei C-S, Chiang K-J and Jung T-P 2020 Facilitating Calibration in High-Speed BCI Spellers via Leveraging Cross-Device Shared Latent Responses *IEEE Trans. Biomed. Eng.* **67** 1105–13

[16] Yger F, Berar M and Lotte F 2017 Riemannian Approaches in Brain-Computer Interfaces: A Review *IEEE Trans. Neural Syst. Rehabil. Eng.* **25** 1753–62

[17] Raza H, Rathee D, Zhou S-M, Cecotti H and Prasad G 2019 Covariate shift estimation based adaptive ensemble learning for handling non-stationarity in motor imagery related EEG-based brain-computer interface *Neurocomputing* **343** 154–66

[18] Saha S and Baumert M 2020 Intra- and Inter-subject Variability in EEG-Based Sensorimotor Brain Computer Interface: A Review *Front. Comput. Neurosci.* **13** 87
[19] Lotte F, Bougrain L, Cichocki A, Clerc M, Congedo M, Rakotomamonjy A and Yger F 2018 A review of classification algorithms for EEG-based brain–computer interfaces: a 10 year update *Journal of Neural Engineering* **15** 031005
[20] Demiralp T, Ademoglu A, Istefanopulos Y, Başar-Eroglu C and Başar E 2001 Wavelet analysis of oddball P300 *International Journal of Psychophysiology* **39** 221–7
[21] Bostanov V and Kotchoubey B 2006 The t-CWT: A new ERP detection and quantification method based on the continuous wavelet transform and Student's t-statistics *Clinical Neurophysiology* **117** 2627–44
[22] Agapov S N, Bulanov V A, Zakharov A V and Sergeeva M S 2016 Wavelet algorithm for the identification of P300 ERP component *arXiv:1611.00033 [q-bio]*
[23] Korczowski L, Congedo M and Jutten C 2015 Single-trial classification of multi-user P300-based Brain-Computer Interface using riemannian geometry *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* 2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC) (Milan: IEEE) pp 1769–72
[24] Simões M, Amaral C, França F, Carvalho P and Castelo-Branco M 2019 Applying Weightless Neural Networks to a P300-Based Brain-Computer Interface *World Congress on Medical Physics and Biomedical Engineering 2018* IFMBE Proceedings vol 68/3, ed L Lhotska, L Sukupova, I Lacković and G S Ibbott (Singapore: Springer Singapore) pp 113–7
[25] LeCun Y, Bengio Y and Hinton G 2015 Deep learning *nature* **521** 436
[26] Craik A, He Y and Contreras-Vidal J L 2019 Deep learning for electroencephalogram (EEG) classification tasks: a review *J. Neural Eng.* **16** 031001
[27] Lawhern V J, Solon A J, Waytowich N R, Gordon S M, Hung C P and Lance B J 2018 EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces *Journal of Neural Engineering* **15** 056013
[28] Zhang X, Yao L, Wang X, Monaghan J, Mcalpine D and Zhang Y 2020 A Survey on Deep Learning-based Non-Invasive Brain Signals:Recent Advances and New Frontiers *arXiv:1905.04149 [cs, eess, q-bio]*
[29] Rakotomamonjy A and Guigue V 2008 BCI Competition III: Dataset II- Ensemble of SVMs for BCI P300 Speller *IEEE Trans. Biomed. Eng.* **55** 1147–54
[30] Sajda P, Gerson A, Muller K-R, Blankertz B and Parra L 2003 A data analysis competition to evaluate machine learning algorithms for use in brain-computer interfaces *IEEE Trans. Neural Syst. Rehabil. Eng.* **11** 184–5
[31] Blankertz B, Muller K-R, Curio G, Vaughan T M, Schalk G, Wolpaw J R, Schlogl A, Neuper C, Pfurtscheller G, Hinterberger T, Schroder M and Birbaumer N 2004 The BCI Competition 2003: Progress and Perspectives in Detection and Discrimination of EEG Single Trials *IEEE Trans. Biomed. Eng.* **51** 1044–51
[32] Blankertz B, Muller K R, Krusienski D J, Schalk G, Wolpaw J R, Schlogl A, Pfurtscheller G, Millan J D R, Schroder M and Birbaumer N 2006 The BCI Competition III: Validating Alternative Approaches to Actual BCI Problems *IEEE Trans. Neural Syst. Rehabil. Eng.* **14** 153–9
[33] Tangermann M, Müller K-R, Aertsen A, Birbaumer N, Braun C, Brunner C, Leeb R, Mehring C, Miller K J, Mueller-Putz G, and others 2012 Review of the BCI competition IV *Frontiers in neuroscience* **6** 55
[34] Lotte F, Congedo M, Lécuyer A, Lamarche F and Arnaldi B 2007 A review of classification algorithms for EEG-based brain–computer interfaces *J. Neural Eng.* **4** R1–13

[35] Amaral C, Mouga S, Simões M, Pereira H C, Bernardino I, Quental H, Playle R, McNamara R, Oliveira G and Castelo-Branco M 2018 A Feasibility Clinical Trial to Improve Social Attention in Autistic Spectrum Disorder (ASD) Using a Brain Computer Interface *Front. Neurosci.* **12** 477

[36] Adamson L B, Bakeman R, Suma K and Robins D L 2019 An Expanded View of Joint Attention: Skill, Engagement, and Language in Typical Development and Autism *Child Dev* **90**

[37] Borra D, Fantozzi S and Magosso E 2020 Convolutional Neural Network for a P300 Brain-Computer Interface to Improve Social Attention in Autistic Spectrum Disorder *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1837–43

[38] Santamaría-Vázquez E, Martínez-Cagigal V, Gomez-Pilar J and Hornero R 2020 Deep Learning Architecture Based on the Combination of Convolutional and Recurrent Layers for ERP-Based Brain-Computer Interfaces *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1844–52

[39] de Arancibia L, Sánchez-González P, Gómez E J, Hernando M E and Oropesa I 2020 Linear vs Nonlinear Classification of Social Joint Attention in Autism Using VR P300-Based Brain Computer Interfaces *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1869–74

[40] Bittencourt-Villalpando M and Maurits N M 2020 Linear SVM Algorithm Optimization for an EEG-Based Brain-Computer Interface Used by High Functioning Autism Spectrum Disorder Participants *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1875–84

[41] Krzemiński D, Michelmann S, Treder M and Santamaria L 2020 Classification of P300 Component Using a Riemannian Ensemble Approach *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1885–9

[42] Miladinović A, Ajčević M, Battaglini P P, Silveri G, Ciacchi G, Morra G, Jarmolowska J and Accardo A 2020 Slow Cortical Potential BCI Classification Using Sparse Variational Bayesian Logistic Regression with Automatic Relevance Determination *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* IFMBE Proceedings vol 76, ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1853–60

[43] Chatterjee B, Palaniappan R and Gupta C N 2020 Performance Evaluation of Manifold Algorithms on a P300 Paradigm Based Online BCI Dataset *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* IFMBE Proceedings vol 76, ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1894–8

[44] Adama V S, Schindler B and Schmid T 2020 Using Time Domain and Pearson's Correlation to Predict Attention Focus in Autistic Spectrum Disorder from EEG P300 Components *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* IFMBE Proceedings vol 76, ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1890–3

[45] Zhao H, Yu S, Prinable J, McEwan A and Karlsson P 2020 A Feasible Classification Algorithm for Event-Related Potential (ERP) Based Brain-Computer-Interface (BCI) from IFMBE Scientific Challenge Dataset *XV Mediterranean Conference on Medical and*

Biological Engineering and Computing – MEDICON 2019 IFMBE Proceedings vol 76, ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1861–8
[46] Manor R and Geva A B 2015 Convolutional Neural Network for Multi-Category Rapid Serial Visual Presentation BCI *Frontiers in Computational Neuroscience* **9** 146
[47] Bittencourt-Villalpando M and Maurits N M 2018 Stimuli and Feature Extraction Algorithms for Brain-Computer Interfaces: A Systematic Comparison *IEEE Trans. Neural Syst. Rehabil. Eng.* **26** 1669–79
[48] Barachant A, Bonnet S, Congedo M and Jutten C 2010 Riemannian geometry applied to BCI classification *International conference on latent variable analysis and signal separation* (Springer) pp 629–36
[49] Drugowitsch J 2019 Variational Bayesian inference for linear and logistic regression *arXiv:1310.5438 [stat]*
[50] Bishop C M 2016 *Pattern Recognition and Machine Learning* (New York, NY: Springer New York)
[51] MacKay D J C 1992 Bayesian Interpolation *Maximum Entropy and Bayesian Methods* ed C R Smith, G J Erickson and P O Neudorfer (Dordrecht: Springer Netherlands) pp 39–66
[52] Benjamini Y and Hochberg Y 1995 Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing *Journal of the Royal Statistical Society. Series B (Methodological)* **57** 289–300
[53] Krusienski D J, Sellers E W, Cabestaing F, Bayoudh S, McFarland D J, Vaughan T M and Wolpaw J R 2006 A comparison of classification techniques for the P300 Speller *J. Neural Eng.* **3** 299–305
[54] Chollet F 2016 Xception: Deep Learning with Depthwise Separable Convolutions *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 1800–7
[55] Borra D, Fantozzi S and Magosso E 2020 Interpretable and lightweight convolutional neural network for EEG decoding: Application to movement execution and imagination *Neural Networks* **129** 55–74
[56] Zhao D, Tang F, Si B and Feng X 2019 Learning joint space–time–frequency features for EEG decoding on small labeled data *Neural Networks* **114** 67–77

CHAPTER 2: A CNN FOR P300 DECODING AND ANALYSIS IN THE SPATIO-TEMPORAL DOMAIN

The study reported in this chapter refers to the published journal paper entitled "Deep learning-based EEG analysis: investigating P3 ERP components" D. Borra and E. Magosso, *Journal of Integrative Neuroscience*, 2021. In this chapter, the CNN structure that resulted the best in the comparison analysis of Chapter 1 was used, not only to decode the P300 but also to design a CNN-based analysis tool of the P300 response. This analysis was based on an input explanation technique to highlight and enhance the relevant neural signatures related to P300 in the spatio-temporal domain.

The neural processing of incoming stimuli can be analyzed from the electroencephalogram (EEG) through event-related potentials (ERPs). The P3 component is largely investigated as it represents an important psychophysiological marker of psychiatric disorders. This is composed by several subcomponents, such as P3a and P3b, reflecting distinct but interrelated sensory and cognitive processes of incoming stimuli. Due to the low EEG signal-to-noise-ratio, ERPs emerge only after an averaging procedure across trials and subjects. Thus, this canonical ERP analysis lacks in the ability to highlight EEG neural signatures at the level of single-subject and single-trial. In this study, a deep learning-based workflow is investigated to enhance EEG neural signatures related to P3 subcomponents already at single-subject and at single-trial level. This was based on the combination of a convolutional neural network (CNN) with an explanation technique (ET). The CNN was trained using two different strategies to produce saliency representations enhancing signatures shared across subjects or more specific for each subject and trial. Cross-subject saliency representations matched the signatures already emerging from ERPs, i.e., P3a and P3b-related activity within 350-400 ms (frontal sites) and 400-650 ms (parietal sites) post-stimulus, validating the CNN+ET respect to canonical ERP analysis. Single-subject and single-trial saliency representations enhanced P3 signatures already at the single-trial scale, while EEG-derived representations at single-subject and single-trial level provided no or only mildly evident signatures. Empowering the analysis of P3 modulations at single-subject and at single-trial level, CNN+ET could be useful to provide insights about neural processes linking sensory stimulation, cognition and behavior.

2.1. INTRODUCTION

Event-related potentials (ERPs) are small changes in the electroencephalogram (EEG), time-locked to a stimulus or an event and reflecting the underlying neural information processing [1]. Thanks to the high-temporal resolution of EEG methodology, analysis of ERPs allows neural processing of an incoming stimulus to be assessed at different stages: from earlier stages, reflected by short-latency (<200 ms post-stimulus) ERP components and mainly mirroring early sensory processing and passive experience, to later stages reflected by long-latency (>250 ms post-stimulus) components and involving cognitive processing of the stimulus such as stimulus evaluation and decision-making processes [2]. Since their discovery, ERPs have been largely used to provide insights into the neural mechanisms underlying sensation, cognition and behavior and have been considered as potential biological markers of neurological and neurodevelopmental disorders [3]. In particular, by comparing ERPs from neurological patients with those of matched healthy controls, steps forward have been made to elucidate impairments in neural processes potentially underlying the investigated psychopathological behavior [4].

Among ERP components, P300 has gaining increasing interest in the last 50 years, since this component plays an important role as psychophysiological marker of psychiatric disorders, such as schizophrenia, bipolar disorder, autism spectrum disorder and depression [5–8], and can also be used as control signal for Brain-Computer Interfaces [9]. The P300 response is an attention-dependent ERP that was first reported in EEG signals by Sutton et al. [10]. This response is characterized by a positive deflection and can be evoked in an oddball task [11], where infrequent deviant (or target) stimuli are presented to the subject immersed in a sequence of frequent background (or standard) stimuli (two-stimulus oddball task, representing the traditional oddball task). The subject attends to stimuli by responding to targets either mentally (e.g., by counting target stimuli) or physically (e.g., by pressing a button when the target stimulus occur), while ignoring other stimuli. Then, the P300 response can be analyzed in the elicited ERPs and is characterized by a wave peaking within the time window between 250 and 500 ms after the stimulus onset and it is mostly distributed on the scalp around the midline EEG electrodes - Fz, Cz, Pz - increasing its magnitude from the frontal to the parietal sites [12]. The oddball P300 wave has been consistently related to attention processes, memory and contextual updating, and decision making [12,13].

Based on results obtained while changing eliciting conditions and stimulus properties [14,15], evidence has emerged that, rather than a single entity, the P300 could be modelled as a “late positive complex”, consisting of different positive subcomponents. In particular, at least two main subcomponents can be distinguished, in part temporally overlapped, namely P3a and P3b which have been associated to distinct, although interrelated, neural processes [12]. P3a is mostly distributed around the midline fronto-central electrodes [12] and is thought to be the marker of orientation of attention [16]. Indeed, P3a has been associated to initial reallocation of attention resulting from the detection of attribute changes in rare stimuli compared to standard ones [12,17]. Moreover, findings suggest a relationship between stimulus deviance and P3a response, that is the greater the mismatch the larger the P3a amplitude [17,18]. Neural sources of P3a seem to be localized in frontal structures and anterior cingulate cortex. P3b has a more posterior-parietal scalp distribution, and longer latency (by 50-100 ms) compared to

P3a. It is assumed to be generated by temporal/parietal structures and to reflect the match between the stimulus and voluntarily maintained attentional trace, relevant for the task at hand, involving memory processes and context updating. According to the neuropsychological model of Polich [12], P3a and P3b reflect two cascade processes, with P3a reflecting attention engagement driven by deviant stimuli initiated in frontal structures and P3b linked to the later phase of task-related stimulus meaning evaluation and working memory comparison.

Due to the difficulty of clearly distinguishing these two components in the traditional two-stimuli oddball task, a modification of this task, resulting in a three-stimulus oddball task, is often used to elicit and investigate these two subcomponents. This paradigm is obtained by inserting a rare non-target stimulus (distractor or novel stimulus) into the sequence of rare targets and frequent standard stimuli, allowing clearly distinguishable P3a and P3b to be obtained in response to the distractor stimuli and target stimuli, respectively. In particular, being the mismatch with the target stimulus larger for the distractor than for the target, the elicited P3a is more evident in the ERPs to distractors than to targets; on the contrary, target ERPs contain a more evident P3b component than distractor ERPs, as only targets are task-relevant stimuli. Investigating both P3b and P3a components is of high interest to study cognitive functions, as they can contribute to better characterize distinct neural subprocesses and may also better discriminate between healthy and pathological conditions; for example P3a has been shown to be more sensitive than P3b in Parkinson's disease [19], depression [20], alcoholism [21] and psychosis [22].

EEG signals are inherently noisy; thus, ERP components emerge only after averaging across trials and subjects (grand averaging procedure), which is the canonical ERP analysis derived from EEG. Therefore, these components and their interpretation may not hold at the subject and/or trial level [23,24]. Thus, investigating component peaks after the grand averaging procedure may hinder the ability to detect and investigate EEG features at the level of single subject or single trial, and consequently, may limit the assessment of relationships between these features and behavior [25] and the assessment of meaningful variability across subjects and even across trials within the same subject. To overcome this limitation, EEG features can be derived without relying on canonical analysis based on ERP component peaks. To this aim, time-frequency decomposition and data-driven approaches, such as machine learning and deep learning algorithms, may represent useful processing steps to obtain reliable estimates of EEG features at the level of single-subject and single-trial, improving the capability to functionally relate EEG features to behavioral performance [25].

In particular, deep learning algorithms - representing a branch of machine learning techniques - consist of computational models designed by stacking layers of artificial neurons (deep neural networks) learning hierarchical feature representations of the input signals via multiple levels of abstraction; that is, deep neural networks learn complex non-linear functions that map inputs to feature representations. In the last decade, deep learning has gained large popularity in fields such as computer vision, speech recognition and natural language processing, to process and classify complex data such as images and time series [26]. Recently, deep neural networks have been started to be explored also with EEG, mainly for classification purposes e.g., to discriminate among trials corresponding to different conditions during a given task [27]. The most common deep learning approach for EEG classification utilizes convolutional neural networks (CNNs) [24–26][28]. These are specialized feed-forward neural

networks including convolution operators at least in one layer and are inspired by the hierarchical structure of the ventral stream of the visual system. In CNNs, neurons with specific local receptive fields are stacked on top of others; thus, receptive fields of neurons increase with the network depth and learned features increase in complexity and abstraction [29]. When CNN is trained in a supervised manner (e.g., in classification), it automatically learns classification-relevant features from the input EEG signals (i.e., class-discriminative features) based on labelled input examples (training stage), and then exploits this learning to classify previously unseen inputs (inference stage). Importantly, CNNs can be fed with raw input signals; therefore, these algorithms are capable of exploiting the entire temporal and spatial information contained in the EEG signal to extract the most class-discriminative features. This represents an important advantage compared to other more traditional machine learning techniques (mainly based on linear discriminant analysis, support vector machines, Riemannian geometry [30,31]) that can handle only limited aspects and/or timepoints of the EEG data [32], thus, not accounting for the overall EEG information; hence, relevant features (and the underlying neural processes) may be ignored in these approaches. In the last years CNNs have been successfully applied to several EEG classification problems, such as the classification of motor activity both imagined and executed [33–37], classification of emotions [38,39] and seizure detection [40]; furthermore, CNNs have found large application to detect the P300 event from single EEG trials [31,33,41–45], also in the perspective of use these algorithms inside Brain-Computer Interface (BCI) systems [9].

Crucially, CNNs not only represent powerful tools for EEG classification, but may also provide novel approaches to improve EEG analysis and interpretation, in particular by exploiting post-hoc (i.e., applied after the training stage) explanation techniques (ET). These are techniques aimed at explaining the features learned by the CNN and that the CNN mostly relies on to discriminate among the classes [46]. Due to the automatic feature learning provided by the CNN, the composition CNN+ET represents a useful non-linear tool to explore the neural processes involved in the classified conditions in a data-driven manner, possibly contributing to validate and also inform cognitive neuroscience knowledge. It is noteworthy that, depending on the training strategy adopted for the CNN (e.g., using signals collected across subjects or signals within single subjects), the features learned by the CNNs may evidence common neural signatures across subjects (representing general task-relevant features), or may evidence neural signatures subject-specific and variability among subjects. Among ETs, saliency maps [47] outline the features within each single input EEG trial that mostly contribute to drive the correct decision (i.e., the correct output class) in the trained CNN; hence, saliency maps represent the timepoints and channels in each EEG input trial that are more relevant for the correct classification of that input example. Since this technique outlines the relevant features in the domain of input EEG signals (which represents a directly interpretable domain), saliency maps can be easily put in relation with ERP correlates. In addition, being the classification performed at the single trial level, saliency maps outline EEG features at the time scale of single trial.

The aim of the present study is to go behind the simple application of CNNs for P3 decoding – as already amply investigated in literature [31,33,41–45] – but rather to explore the potentialities of the combination CNN+ET as a data-driven EEG analysis tool in the investigation of the electrophysiological signatures related to P3 (in particular, to its main subcomponents P3a and P3b), and to test its ability to enhance relevant signatures already at

the level of single-subject and single-trial. Therefore, the novelty of this study is the formalization of a procedure CNN+ET useful for analyzing meaningful features in EEG signals in response to events, and complementary to (and potentially more powerful than) the more classical ERP analysis. To this aim, we used a CNN to classify, at the level of single-trial, the EEG responses to target, distractor and standard stimuli in a 3-stimulus oddball task collected on several subjects; the CNN was realized using EEGNet, a previously validated CNN for P300 decoding [33]. Two different training strategies were adopted, training CNNs using EEG trials from all subjects and from single subjects, so that the obtained classifiers could learn common cross-subject and subject-specific class-discriminative features, respectively. Then, saliency maps were used as ET and were applied to the target and distractor classes, to highlight the spatio-temporal samples of the input that resulted more class-discriminative, potentially highlighting P3b- and P3a-related features. Three different levels of representations and analyses were possible with this approach: cross-subject, within-subject and single-trial. The contribution of this study is twofold:

i. Test the capability of a CNN to discriminate trials in a 3-stimulus oddball tasks, automatically identifying features in the input data that correspond to relevant characteristics of the ERP response (such as different proportions of P3a and P3b manifestations), by using CNN-derived representations at the cross-subject level.
ii. Investigate how the adopted CNN+ET combination may enhance relevant neural signatures underlying the task at hand, both at the level of single subject and of single trial, overcoming the limitation of the canonical ERP analysis derived from a grand averaging procedure over EEG trials.

2.2. MATERIALS AND METHODS

In this section, first we present the three-stimulus oddball dataset. Then, we formalize the problem of decoding the EEG signals of the dataset via CNNs and how this approach, coupled with an ET, could be used as an analysis tool. Subsequently, the specific CNN architecture and training strategies adopted here are illustrated; finally, the computation of the saliency maps, used as explanation technique, is described.

2.2.1. Dataset and pre-processing

In this study we adopted a public dataset [48] (available at https://openneuro.org/datasets/ds003490/versions/1.1.0) consisting of EEG signals recorded from 64 electrodes during a 3-auditory oddball task. Three stimuli were provided to 25 healthy participants for 200 ms using stereo speakers: standard stimuli (70% of trials) were 440 Hz sinusoidal tones, target stimuli (15% of trials) were 660 Hz sinusoidal tones and novel distractors (15% of trials) were sampled from a naturalistic sound dataset [49]. Participants mentally counted the number of target stimuli ignoring standard and distractor stimuli, resulting in a covert response (thus removing motor activity influences). A total number of 200 trials (140, 30 and 30 trials, respectively for standard, target and distractor conditions) were recorded for each participant. EEG was recorded at 500 Hz with reference at CPz and ground at AFz.

In order to be consistent with the reference study by Cavanagh et al [48] where these signals were first collected and analyzed, we decided to adopt here the same pre-processing pipeline as in that previous study, using the same version of the Matlab toolbox EEGlab (version 14_0_0b, Swartz Center for Computational Neuroscience, UC San Diego, CA, USA) [50]. The processing steps are described below:

i. Removal of the very ventral electrode signals (FT9, FT10, TP9, TP10) as they tend to be unreliable.
ii. Epoching between [-2,2] s respect to stimulus onset.
iii. Re-referencing to an average reference to recover CPz activity.
iv. Identification of bad channels. To this aim, channels were separately marked for rejection computing the kurtosis of each channel finding outliers (default method used in the EEGLAB function “pop_rejchan” to perform automatic channel rejection), and applying the FASTER algorithm [51]. Channels that were automatically labelled for rejection from both previous algorithms were then rejected. Finally, bad channels were interpolated using spherical interpolation.
v. Bad epochs were marked using the FASTER algorithm [51] and then removed. After this step, the average number of trials per participant was reduced to 130±2, 29±1, 29±1 (mean ± standard deviation across participants), respectively for standard, target and distractor conditions.
vi. Removal of independent components related to eye blinks.
vii. Re-referencing to an average reference.
viii. Baseline correction from -0.2 to 0 s pre-stimulus.
ix. Band-pass filtering between 0.1-20 Hz. This filtering was included in the pre-processing pipeline of [42]; furthermore, it is worth noticing that this kind of filtering was performed

also in other CNN-based P3 decoding studies [31,42,43,45] (however, as reported in the Discussion, we also tested the effect on CNNs performances of maintaining a large frequency content of the signals, between 0.1-40 Hz).

In addition to these steps, to reduce the size of the input in the CNN-based decoding, we downsampled signals to 100 Hz and considered EEG in epochs between [0,1] s post-stimulus. These steps reduced the time samples of the input to be processed in the CNN-based decoder. Thus, after this pre-processing pipeline, each EEG trial was a 2D matrix of shape $(C, T) = (60,100)$, where C represents the number of spatial channels (electrodes) and T the number of time steps.

2.2.2. CNN-based EEG decoding and analysis

The EEG dataset of each subject participating in the study consisted of separated pre-processed trials (see Section 2.1) with each trial belonging to one of the conditions of interest, i.e., standard, target and distractor. Each subject-specific dataset can be denoted by:

$$D^{(s)} = \{(X_0{}^{(s)}, y_0{}^{(s)}), \dots, (X_i{}^{(s)}, y_i{}^{(s)}), \dots, (X_{M^{(s)}-1}{}^{(s)}, y_{M^{(s)}-1}{}^{(s)})\}, \quad (2.1)$$

indicating with $M^{(s)}$ the number of trials of the s-th subject ($0 \leq s \leq N-1$ where N is the number of subjects).

$X_i{}^{(s)}$ is composed by the pre-processed EEG signals of the i-th trial, while $y_i{}^{(s)}$ is its associated label:

$$\begin{cases} X_i{}^{(s)} \in \mathbb{R}^{C \times T}, 0 \leq i \leq M^{(s)} - 1 \\ y_i{}^{(s)} \in L = \{l_0, l_1, l_2\} = \{\text{"standard"}, \text{"target"}, \text{"distractor"}\} \end{cases}. \quad (2.2)$$

A CNN can be trained to realize a classifier f aimed to discriminate between these 3 conditions (3 output classes). In this supervised learning framework, during a training stage the system automatically learns, from a training set of EEG trials, the more relevant features for a correct classification, so that it can subsequently assign the correct class label to new unseen trials (belongings to the test set). That is, the CNN describes the function f:

$$f(X_i{}^{(s)}; \theta): \mathbb{R}^{C \times T} \rightarrow L, \quad (2.3)$$

parametrized in the parameter array θ (whose values are learned during training), mapping a label to each trial $X_i{}^{(s)}$, where $X_i{}^{(s)}$ represents the CNN input (2D matrix of shape (C, T)).

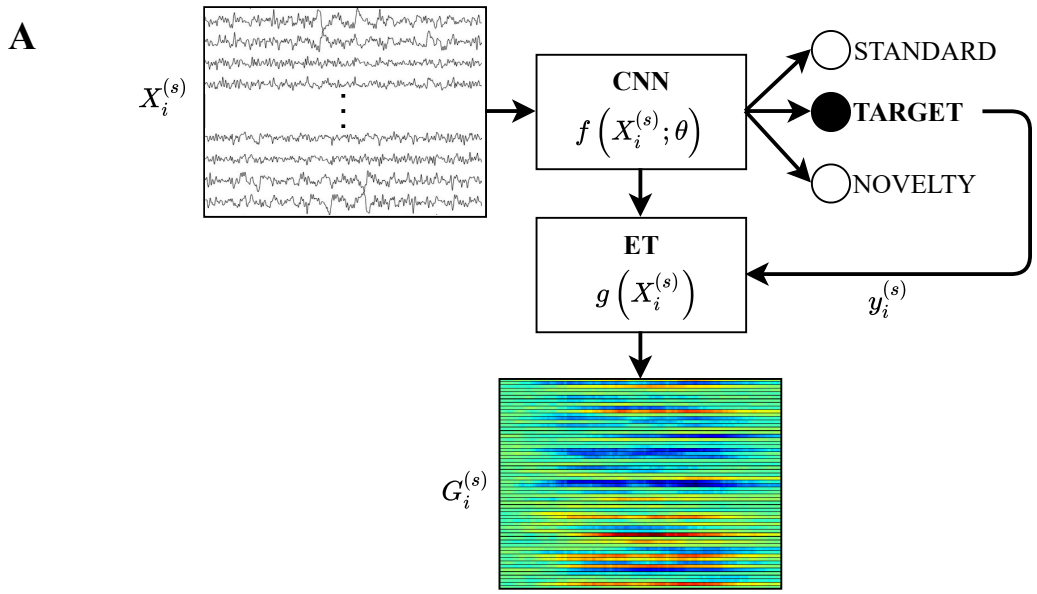
Adopting this 2D representation, CNN inputs preserve the original EEG structure. Going further deeper in the CNN, the algorithm processes the single-trial representation exploiting hierarchically structured features finalized to discriminate among classes (e.g., standard, target or distractor conditions).

Then, the trained CNN processes test trials $X_i{}^{(s)}$ to discriminate between conditions of interest based on the class-discriminative features learned during training. The knowledge behind the discrimination $f(X_i{}^{(s)}; \theta)$ operated by the CNN using the input trial $X_i{}^{(s)}$ could be explained by deriving the most relevant features in that input example that drive the correct classification; in this way, meaningful neural signatures in the EEG input trial can emerge, related to the neural processes underlying the task at hand. To do so, the CNN can be paired with an ET, that computes for each spatio-temporal sample of the input trial $X_i{}^{(s)}$, a relevance score indicating how relevant is that sample for the network to provide the correct classification. Thus, an ET provides a relevance representation g of the input $X_i{}^{(s)}$:

$$g\left(X_i^{(s)}\right): \mathbb{R}^{C\times T} \rightarrow \mathbb{R}^{C\times T}, \tag{2.4}$$

where the function g depends on the trained classifier f, on the ground truth label $y_i^{(s)}$ assigned to the input trial, and on the specific method adopted to produce the relevance (Figure 2.1A).

Therefore, according to this approach, each input trial $X_i^{(s)}$ is processed by CNN+ET exploiting a highly non-linear transformation $g\left(X_i^{(s)}\right)$ aimed to enhance, already at the single trial level, the meaningful information contained in the EEG signals, by highlighting the spatio-temporal samples more discriminative for each condition and likely informative of the neural processing underlying the type of stimulus provided to the subject.



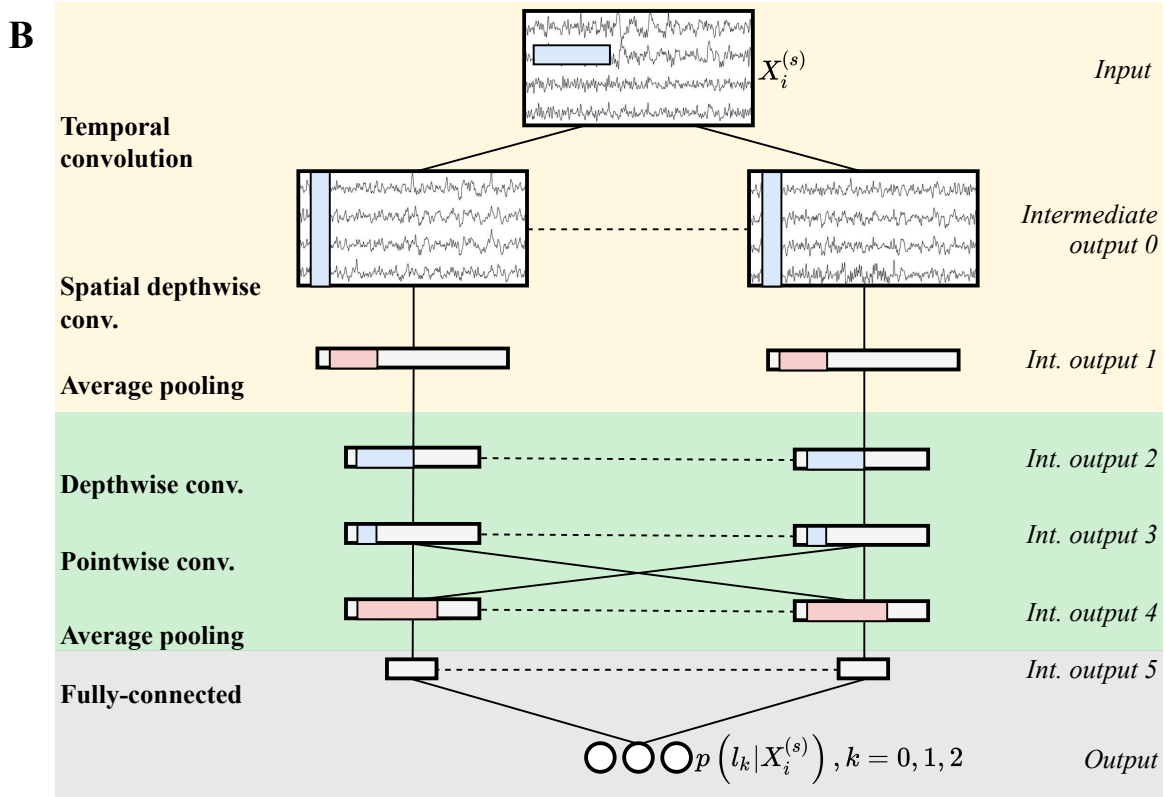


Figure 2.1 – Proposed data-driven EEG analysis tool: workflow and scheme of the adopted CNN. (A) CNN+ET analysis framework. The input EEG trial $X_i^{(s)}$ is processed by the CNN-based parametric classifier $f(X_i^{(s)};\theta)$. Using $f(X_i^{(s)};\theta)$ and the correct output label $y_i^{(s)}$, the relevance representation $g(X_i^{(s)})$ is computed. (B) Schematic representation of EEGNet, implementing the parametric classifier f. Only its main layers are represented, for a more detailed description see Section *CNN design: EEGNet* and Supplementary Table 2.1; in particular note that non-linear activation layers have been omitted in this figure, but they are present in the implementation. The input $X_i^{(s)}$ is processed by the CNN through many layers (with the layer name reported on the left) organized in 3 main blocks (spatio-temporal block: yellow, temporal block: green, fully-connected block: grey) to obtain the output conditional probabilities $p(l_k|X_i^{(s)}), k = 0,1,2$. The intermediate output of each layer is reported as a white box with the spatial and temporal dimensions along rows and columns, respectively. Coloured boxes represent the filters of convolutional (blue boxes) and pooling (red boxes) layers.

2.2.3. Proposed CNN+ET approach

CNN design: EEGNet

In this study we adopted EEGNet [33], a light (in terms of parameters to fit) CNN previously validated to discriminate between target and standard stimuli in a 2-stimuli oddball paradigm. This lightweight design was chosen to reduce the risk of overfitting, as the dataset adopted here consisted of a small number of examples for each subject (see Section 2.1). In particular, EEGNet is the lightest design among others proposed in the literature for P300 decoding [31]; using one of the other available CNNs (introducing >10K trainable parameters), the model would be more prone to overfitting.

EEGNet is composed by 3 main blocks (Figure 2.1B). The first one can be referred to as a spatio-temporal block (yellow block in Figure 2.1B). It first processes the input EEG trial $X_i^{(s)}$ to provide temporal features maps by applying convolutional filters to each single electrode (see intermediate output 0 in Figure 2.1B). In our implementation, 8 temporal filters were learned. Next, spatial filters spanning all the electrodes are learned by applying depthwise convolution, where each spatial filter is applied to just one previous feature map; the number of spatial filters learned for each temporal filter was set to 1 in our implementation (see intermediate output 1 in Figure 2.1B). Then, a layer applying a non-linear activation function (Exponential Linear Unit, ELU) to the spatially filtered activations is employed. This layer is followed by an average pooling layer, to reduce the computational cost; we adopted average pooling over 3-time steps with stride of 3, and these averaged activations are provided to the second block (green block in Figure 2.1B). The second block uses depthwise convolution and pointwise convolution (overall realizing a separable convolution) to summarize the spatially filtered activities (see intermediate output 3 in Figure 2.1B) in the temporal domain; here, separable convolution is designed to learn temporal patterns of about 500 ms on the spatially filtered activations. As in the first block, a subsequent layer applying an ELU activation function was employed, followed by average pooling (in this implementation over 6 time samples with a stride of 6) that further reduces temporal samples. Lastly, these activations were provided to a single fully-connected layer (see Figure 2.1B, grey block) consisting of 3 output neurons activated via a softmax function to produce the output probability distribution. Thus, the CNN output provides the conditional probabilities $p(l_k|X_i^{(s)}), k = 0,1,2$ for the conditions to be discriminated. Further details about the CNN and about its hyper-parameters (i.e., non-

trainable parameters defining the unique functional form of the CNN) are reported in Supplementary Materials (see Section 2.6.1 and Supplementary Table 2.1).

The architecture comprises also layers aimed to increase the generalization of the model (i.e., regularizers), such as batch normalization and dropout layers (with a dropout probability of 0.5, see also Section 2.6.1 of Supplementary Materials and Supplementary Table 2.1 for further details), in addition to regularizers applied during the training phase, such as early stopping. EEGNet hyper-parameters adopted here were different compared to its original formulation [33], as we carefully chose them (see Supplementary Table 2.1) to keep limited the overall number of trainable parameters (consisting of only 1259 parameters) in consideration of the very small dataset handled here, in view of further reducing the risk of overfitting. CNNs were developed in PyTorch [52] and trained using a workstation equipped with an AMD Threadripper 1900X, NVIDIA TITAN V and 32 GB of RAM. Codes will be released at https://github.com/ddavidebb/CNN-based_P3_analysis.git.

Training strategy

The training stage of EEGNet (i.e., optimization of parameters contained in θ) was performed using two different training strategies, at cross-subject level and within-subject level; the former is useful to evidence general EEG signatures common to all subjects, while the latter can emphasize possible differences among the subjects. Specifically, the following strategies were adopted:

i. Leave-one-subject-out (LOSO) strategy. In this approach, the data of the s-th subject were held out and used as test set (thus, the test set corresponds to the entire dataset $D^{(s)}$ of that subject), while the data of all other 24 subjects were used as training set. This procedure was repeated until each subject was selected as test subject; therefore, 25 cross-subject CNNs were obtained, each one "agnostic" about the specific s-th subject used for testing.
ii. Within-subject (WS) strategy. In this approach, for each s-th subject, a CNN was trained and tested using only data for that subject, thus, realizing a subject-specific CNN ($\theta = \theta^{(s)}$). Since the dataset of each subject was limited, each subject-specific dataset $D^{(s)}$ was partitioned into a training set and a test set adopting a 10-fold stratified cross-validation scheme. It is worth noticing that considering all the 10 folds of the cross-validation procedure, all the trials of the dataset $D^{(s)}$ of that subject were tested.

In both cases, a validation set composed by 20% of the training examples was extracted and used to define stop criteria (see early stopping below) for the optimization of the CNN. In WS trainings, the validation set was sampled by keeping the same class proportion as in the training set (i.e., sampling 20% of each class for each subject). In LOSO trainings the validation set was equally sampled from the 24 subjects (by sampling 20% of signals from each participant's training set) and, in this case too, by keeping the same class proportion as in the training set.

The cross-entropy between the empirical probability distribution (defined by training labels) and the model probability distribution (defined by CNN outputs) was used as loss function and was minimized using the Adaptive moment estimation (Adam) algorithm [53] with a mini-batch size of 32, learning rate of 10^{-4} and other parameters set as in its default implementation [52]. CNNs were trained for 500 epochs, early stopping the optimization when the validation

loss did not decrease for 50 consecutive epochs (set on the basis of the convergence speed of the algorithm via empirical evaluations), as also performed previously in [42]. To address class imbalance, parameter updates were weighted more or less depending on the class occurrence of the input examples. In particular, indicating with ${M_0}^{(s)}$, ${M_1}^{(s)}$, ${M_2}^{(s)}$ the number of trials for standard, target and distractor conditions for the generic s-th subject and given that ${M_0}^{(s)} > {M_1}^{(s)}$ and ${M_0}^{(s)} > {M_2}^{(s)}$, class weights were defined as 1, ${M_0}^{(s)}/{M_1}^{(s)}$, ${M_0}^{(s)}/{M_2}^{(s)}$, respectively for standard, target and distractor conditions.

Performance metrics

In this study, we used the Area Under the ROC curve (AUROC) to evaluate the performance of each trained CNN in the 3-class classification task; this metrics is commonly adopted to measure performance of P3 decoding at the level of single trials [33,42,44], a task intrinsically characterized by class imbalance (since P3 is evoked by infrequent stimuli as opposed to frequent ones). In particular, the AUROC (evaluated on the test set for each training strategy) of each possible pairwise combination of classes (one-vs-one, OVO) was computed - i.e., standard vs. target, target vs. distractor, standard vs. distractor - and then averaged across these three combinations, obtaining a multi-class AUROC (referred as av-AUROC in this study) [54]. Furthermore, we computed also the F1 score and Area Under the Precision Recall curve (AUPR) for the classes whose neural signatures were investigated in this study (i.e., target and distractor conditions), and these further metrics are reported in the Supplementary Materials (Supplementary Table 2.2).

Statistical analysis

To compare the OVO AUROCs and av-AUROCs between WS and LOSO strategies, Wilcoxon signed-rank tests were performed. To correct for multiple tests (4 in total), a false discovery rate correction at α = 0.05 using the Benjamini-Hochberg procedure [55] was applied.

Explanation technique: saliency maps computation and processing

Once networks were trained adopting WS and LOSO strategies, a post-hoc ET was used to derive useful representations about input spatio-temporal samples contributing more to the discrimination of target and distractor classes to investigate P3b- and P3a-related correlates. Here we adopted saliency maps [47] to compute the relevance score of each sample belonging to the input layer (overall $C \cdot T$ samples of the single-trial EEG data) for a specific class decision. It is useful to remember that both in case of LOSO and WS strategies, the entire dataset $D^{(s)}$ of each subject was tested (see Section *Training strategy*); in the first strategy, using a cross-subject CNN (trained on the other subjects), in the second case using subject-specific CNNs (trained in a cross-validation scheme).

For each input trial ${X_i}^{(s)}$ belonging to the test set of the s-th subject, the relevance ${G_i}^{(s)} = g\left({X_i}^{(s)}\right)$ was computed by backpropagating the gradient of the output neuron corresponding to the correct label ${y_i}^{(s)}$ (representing the output activation, immediately before the softmax function) back to the input layer (see Figure 2 for a schematic representation of the saliency

map computation and processing). The relevance map had the same shape as the input (${G_i}^{(s)} \in \mathbb{R}^{C\times T}$) and each ${G_{i,jk}}^{(s)}$ sample (indicating with j and k rows and columns, respectively) quantified how much a variation in the corresponding jk sample of the single trial, i.e., ${X_{i,jk}}^{(s)}$, affected the activation of the correct output neuron. That is, for each subject's dataset $D^{(s)}$, the associated collection of relevance was given by:

$$G^{(s)} = \{\left({G_0}^{(s)}, {y_0}^{(s)}\right), \dots, \left({G_i}^{(s)}, {y_i}^{(s)}\right), \dots, \left({G_{M^{(s)}-1}}^{(s)}, {y_{M^{(s)}-1}}^{(s)}\right)\}, \quad (2.5)$$

containing one saliency map paired to each input trial. Then, these 2D trial-specific saliency maps were averaged across trials belonging to target and distractor classes; in this way, a saliency map for each output class was obtained for each subject. No post-processing was applied to the so obtained maps (e.g., computing the absolute or the square value), preserving the entire information. However, as the investigated EEG correlates involve positive modulations in ERPs respect to the standard condition (i.e., P3a and P3b), in this study we focused on positive values of the saliency maps, i.e., positive (negative) perturbations of input samples that increased (decreased) the correct class score.

The previous procedure was applied both to cross-subject CNNs obtained with the LOSO strategy and to subject-specific CNNs obtained with the WS strategy, resulting in LOSO and WS saliency maps, respectively (see the diagram in Figure 2).

In particular, for each subject two saliency maps (corresponding to target and distractor classes) were obtained both for the LOSO and WS strategies. LOSO saliency maps, being obtained from models trained on multiple subjects' distributions, are more likely to reflect optimal class-discriminative input samples that are shared across subjects. Therefore, these representations allowed general task-related class-discriminative input samples to be inspected. Conversely, WS saliency maps – obtained from models trained on subject-specific distributions – are more likely to reflect subject-specific features. Therefore, these representations allowed the investigation of inter-subject variability of the more class-discriminative input samples.

Then, the LOSO and WS saliency maps (two maps for each subject), were subjected to different processing. In the LOSO-CNN+ET analysis pipeline, saliency maps were averaged across subjects, separately for each output class of interest, to obtain a "2D cross-subject saliency map" for each output class. Then, the 2D cross-subject saliency map was also averaged across all electrodes or across all time samples, obtaining a "temporal cross-subject saliency pattern" and a "spatial cross-subject saliency pattern", respectively. In addition, spatial cross-subject patterns were computed averaging the 2D maps only within selected time windows comprising the main peaks of the temporal cross-subject pattern. These representations allowed an analysis at the cross-subject level resulting from a grand average procedure, similarly to ERPs. Conversely, in the WS-CNN+ET analysis pipeline, saliency maps were analyzed separately for each subject. For each subject, the 2D saliency map of each output class was averaged across electrodes or time samples to obtain a "temporal subject-specific saliency pattern" and a "spatial subject-specific saliency pattern", respectively. Then, hierarchical agglomerative clustering (HAC) [56] was performed on temporal subject-specific patterns (separately for each output class of interest), to identify clusters of subjects characterized by different temporal saliency patterns. Different clusters denote different strategies adopted by the CNN in exploiting input samples to discriminate a specific class and may reflect differences across subjects in the underlying neural processes. In particular, HAC was performed using a

complete linkage (i.e., farthest neighbour clustering) and adopting the correlation between observations as distance metric (see Section 2.6.2 of Supplementary Materials for a description of the adopted distance metric). Four clusters were considered and the temporal subject-specific patterns of the subjects within each cluster were averaged to obtain an average temporal saliency pattern at the level of cluster ("temporal cluster-specific saliency pattern"). In addition, the spatial subject-specific patterns of the subjects within each cluster (as resulted from the clustering in the temporal domain) were averaged, to obtain an average spatial saliency pattern at the level of cluster ("spatial cluster-specific saliency pattern").

Finally, we performed an analysis to investigate whether the proposed CNN+ET combination could be useful to enhance correlates related to P3a and P3b at the level of single subject and single trial compared to a canonical analysis based on evoked potentials. To this aim, for each condition of interest (target and distractor), we selected a single subject belonging to each cluster, as representative of that specific cluster, and we visually evaluated to what extent the information contained in the temporal saliency pattern of that subject and condition were contained and already visible in the evoked potentials of that subject for that condition (obtained by averaging the EEG trials of that subject corresponding to that condition). To perform such comparison, for each representative subject selected, evoked potentials were averaged together within a subset of electrodes that showed more P3a and P3b components; the temporal saliency maps to be compared were obtained by averaging the 2D subject-specific saliency maps across the same subset of electrodes too (rather than across all electrodes). These subsets of electrodes were P3, P1, Pz, P2, P4, PO3, POz, PO4 (showing more P3b) for target condition and F1, Fz, F2, FC1, FCz, FC2 (showing more P3a) for distractor condition (for this choice, see ERPs in Section 2.3.1). In this way, the comparison was limited on a small subset of electrodes that more expressed the specific ERP component of interest. Lastly, for each of the previous temporal patterns (i.e., temporal subject-specific saliency pattern and evoked potential, each one averaged across a specific subset of electrodes) that were both obtained by averaging across trials, we considered the single constituent trials and compared the associated saliency at the level of single trial with the corresponding EEG trial, still maintaining the averaging across the specific subset of electrodes.

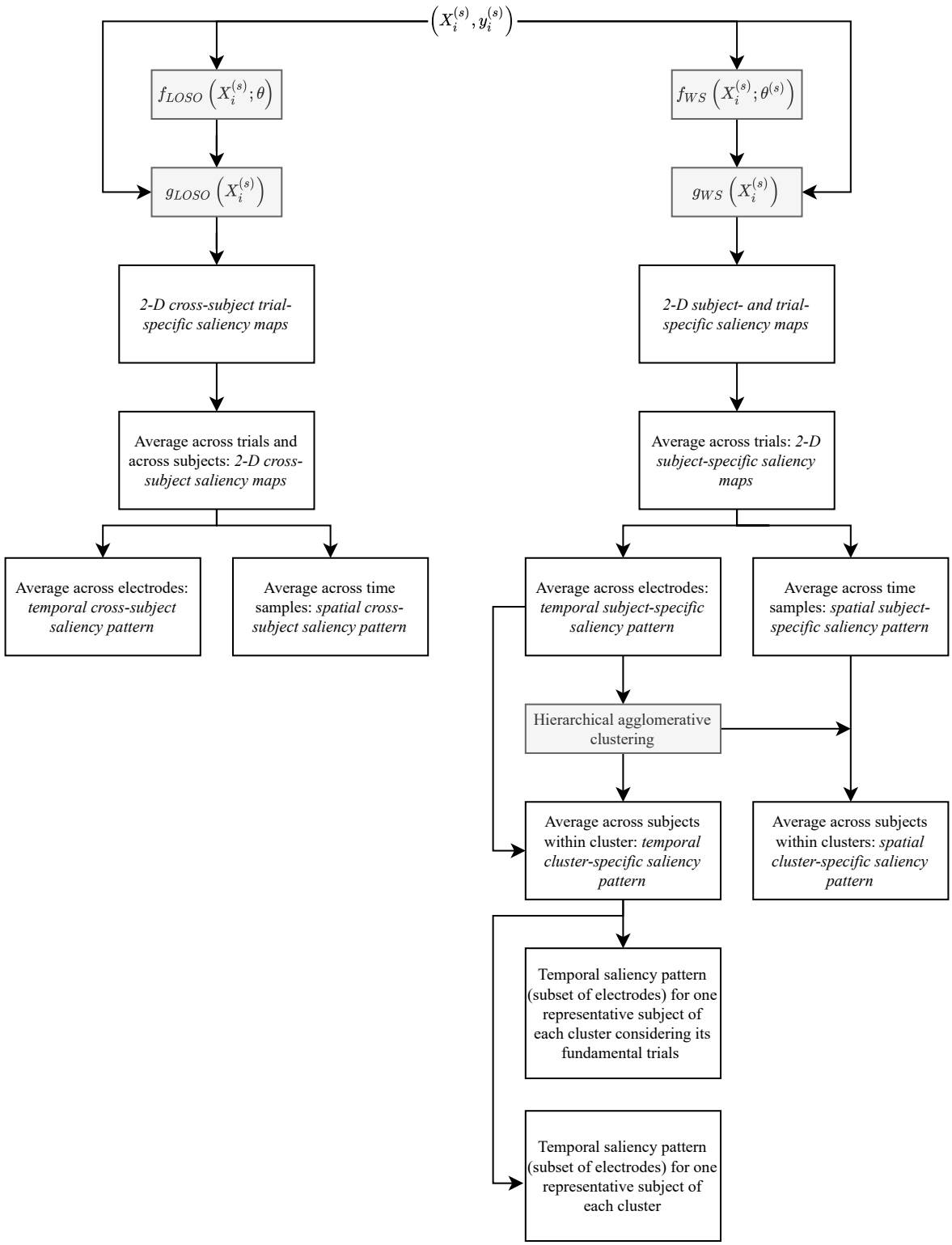


Figure 2.2 – Schematic diagram of the saliency map computation and processing performed on LOSO (left branch, characterized by $f_{LOSO}\left(X_i{}^{(s)};\theta\right)$ and $g_{LOSO}\left(X_i{}^{(s)}\right)$) and WS (right branch, characterized by $f_{WS}\left(X_i{}^{(s)};\theta^{(s)}\right)$ and $g_{WS}\left(X_i{}^{(s)}\right)$) models.

2.3. RESULTS

2.3.1. Event Related Potentials

The grand average ERPs of the target and distractor conditions are reported in Figures 2.3 and 2.4, respectively. The same representations for the standard condition are reported in Supplementary Figure 2.1. These figures represent the conventional grand average across EEG trials (of the same condition) and across subjects, to obtain the evoked potentials.

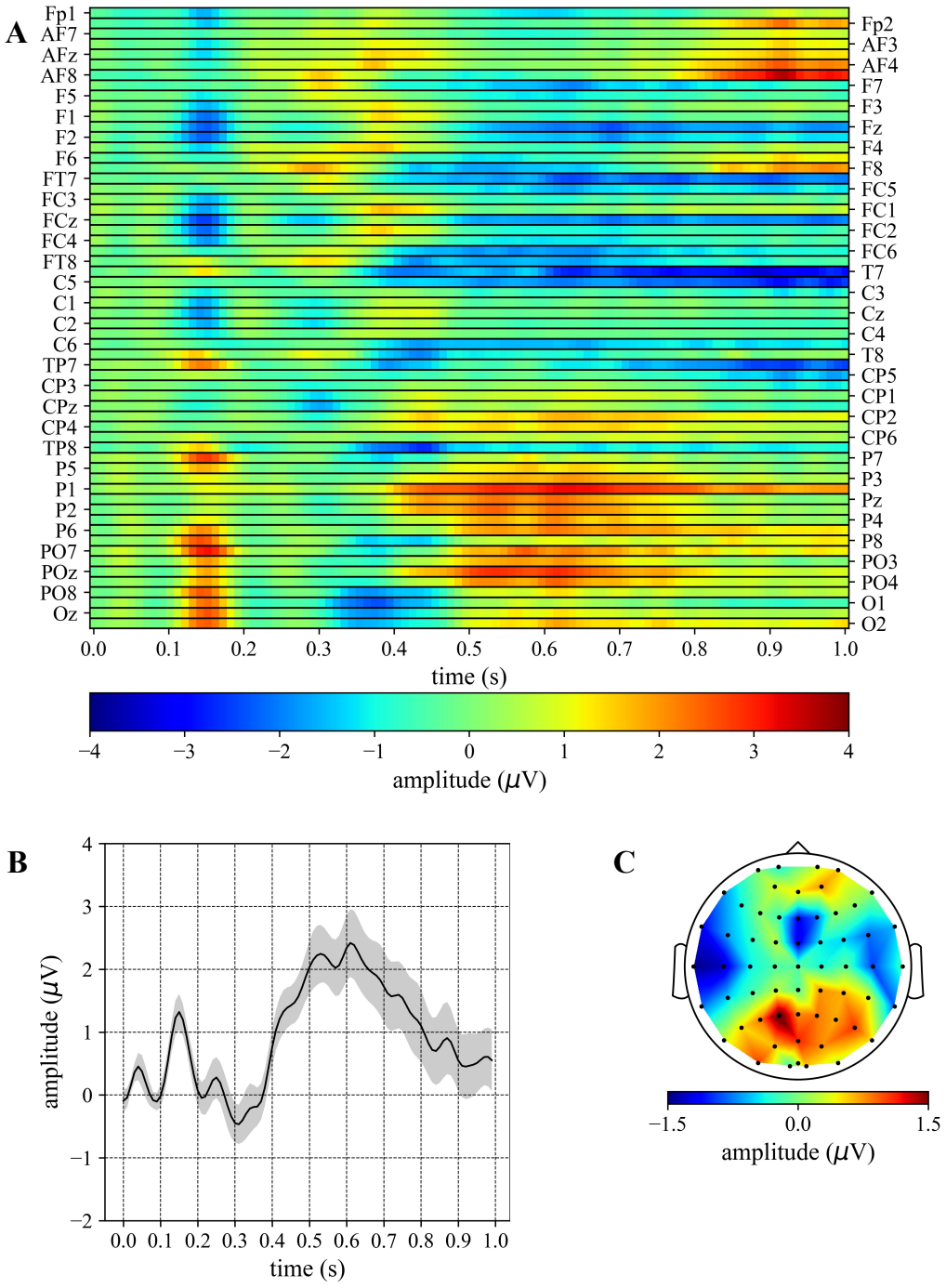


Figure 2.3 – Grand average ERP: target. (A) The grand average is reported as a 2D heatmap with electrodes and time steps along rows and columns, respectively. (B) The average temporal pattern obtained by averaging the 2D heatmap of panel A across the subset of electrodes showing more the P3b subcomponent (P3, P1, Pz, P2, P4, PO3, POz, PO4). The shaded area represents the mean value ± standard error of the mean, while the thick line represents

the mean value. (C) Topological representation of the average contribution of each electrode across all time samples of the 2D heatmap.

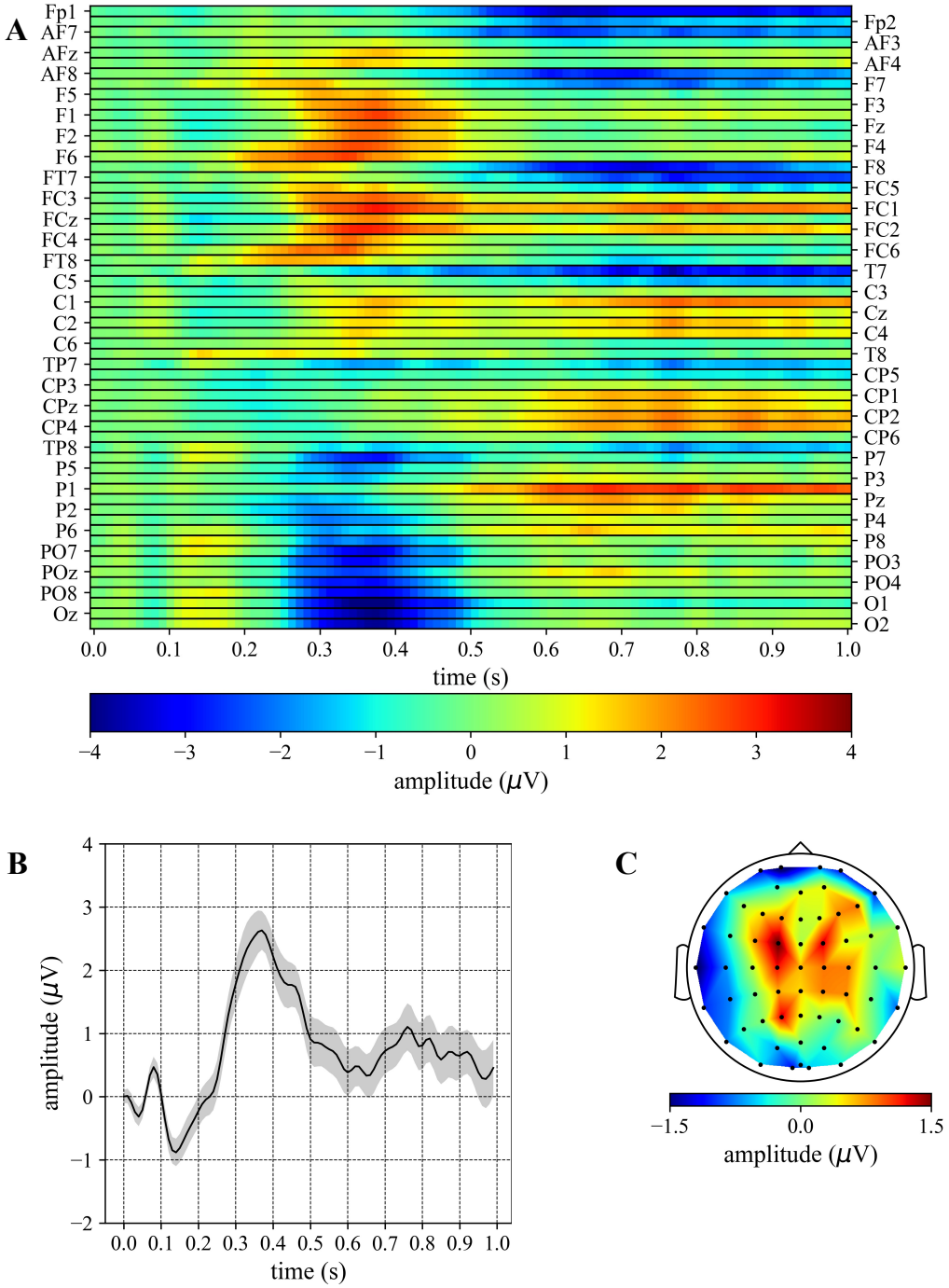


Figure 2.4 – Grand average ERP: distractor. (A) The grand average is reported as a 2D heatmap with electrodes and time steps along rows and columns, respectively. (B) The average temporal pattern obtained by averaging the 2D heatmap of panel A across the subset of electrodes showing more the P3a subcomponent (FC1, FC2, C1, C2, CP1, CP2). The shaded area represents the mean value ± standard error of the mean, while the thick line represents the mean value. (C) Topological representation of the average contribution of each electrode across all time samples of the 2D heatmap.

In particular, grand averages are reported for all electrodes as 2D heatmaps (panels A), showing the main P3 components, P3a and P3b, with different proportions. The P3a component

can be observed especially in fronto-central/frontal regions (e.g., F1, Fz, F2, FC1, FCz, FC2, Figure 2.4A) for the distractor condition while it is less evident in the same regions for the target condition (see Figure 2.3A). In addition, the P3b component can be observed for the target condition in parieto-occipital/parietal regions (e.g., P3, P1, Pz, P2, P4, PO3, POz, PO4, Figure 2.3A), but not for the distractor condition (Figure 2.4A). Lastly, another component with a higher latency can be individuated in the distractor condition from centro-parietal to fronto-central regions (e.g., FC1, FC2, C1, C2, CP1, CP2, Figure 2.4A). Averaging the activity across different subsets of electrodes showing more P3a and P3b, the timing of P3a and P3b components became clearer, i.e., averaging across P3, P1, Pz, P2, P4, PO3, POz, PO4 for the target condition (Figure 2.3B) and across F1, Fz, F2, FC1, FCz, FC2 for the distractor condition (Figure 2.4B) (and standard condition too, Supplementary Figure 2.1B). In particular, the P3a and P3b appeared peaking within the time window 325-375 ms and 500-700 ms, respectively. Lastly, we reported also the overall spatial contribution by averaging the grand average of each electrode across all time samples (panels C), highlighting the overall topology of these components over the entire epoch (0-1000 ms).

2.3.2. CNN performance

At first, the performance of the CNN on the test set in the discrimination between the contrasted conditions needs to be analyzed, to validate the CNN in the objective discrimination task and thus, evaluate whether the CNN learned useful and robust class-discriminative features. This is an important validation stage as successive steps based on the combination CNN+ET exploit features learned by this system to derive useful representations and then to analyze P3 subcomponents (see Section *Explanation technique: saliency maps computation and processing*).

OVO AUROCs (mean ± standard error of the mean across the subjects) are reported in Figure 2.5A, while av-AUROCs for each subject (the average across the three OVO AUROCs, see Section *Performance metrics*) are shown in Figure 2.5B, both for LOSO strategy and WS strategy. Furthermore, F1 scores and AUPRs are reported in the Supplementary Table 2.2. EEGNet scored av-AUROCs of 76.2±1.3% and 70.5±1.2%, respectively for LOSO and WS models. Cross-subject models (as obtained with the LOSO strategy) achieved higher av-AUROCs ($p = 2.1 \cdot 10^{-4}$) respect to subject-specific models (as obtained in the WS strategy). This was primarily related to a significant improvement in the discrimination between standard vs. distractor conditions ($p = 1.7 \cdot 10^{-5}$), while other combinations resulted comparable in performance between the two strategies (see Figure 2.5A). Lastly, subject-level av-AUROCs (Figure 2.5B) were above the value obtained with a random classifier (i.e., $AUROC = 0.5$) in all cases.

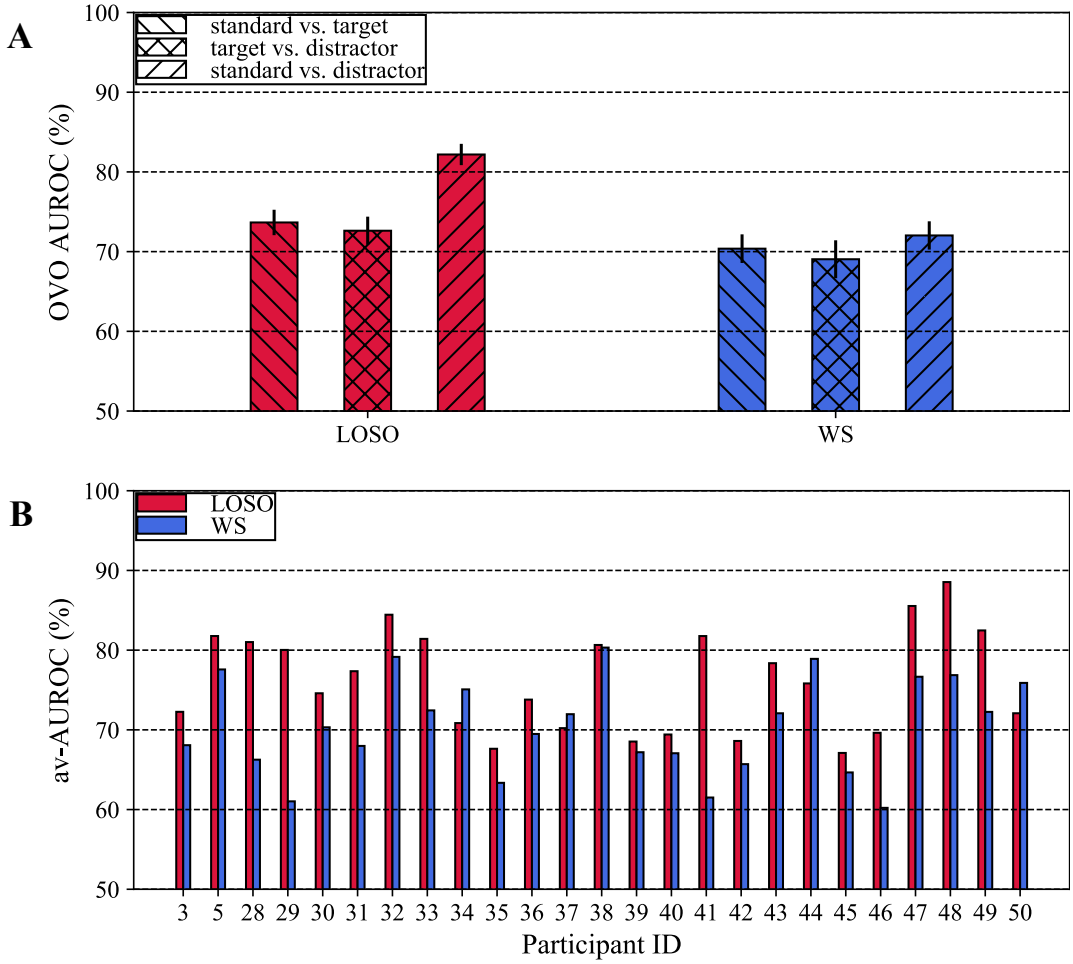


Figure 2.5 – CNN performance. (A) One-vs-one AUROCs using LOSO (red bars) and WS (blue bars) strategies. The height of bars denotes the mean value across subjects, while the error bar denotes the standard error of the mean. (B) Multi-class AUROCs (also referred as av-AUROCs in the manuscript) at the level of single subject obtained with the LOSO (red bars) and WS (blue bars) strategies. Note that the participant ID reported on the x-axis reflects the participant ID of the dataset.

2.3.3. EEG analysis based on the CNN and explanation technique

In this section, the results obtained analyzing EEG signals with the CNN+ET combination are reported. These consisted in relevance representations of the input EEG data that supported more the discrimination between the three contrasted conditions, as operated by EEGNet. In particular, the relevance is reported for target condition and distractor conditions, as the EEG response associated to these stimuli allow the analysis of P3b and P3a.

Cross-subject saliency

In Figure 2.6 and 2.7 the 2D cross-subject saliency maps together with the derived temporal and spatial cross-subject saliency patterns are reported for target condition and distractor condition, respectively. These figures are obtained via the application of the LOSO CNN+ET procedure (see left branch in Figure 2.2); therefore, they are obtained differently from Figures 2.3 and 2.4 which represent the canonical grand average ERP derived directly from EEG trials.

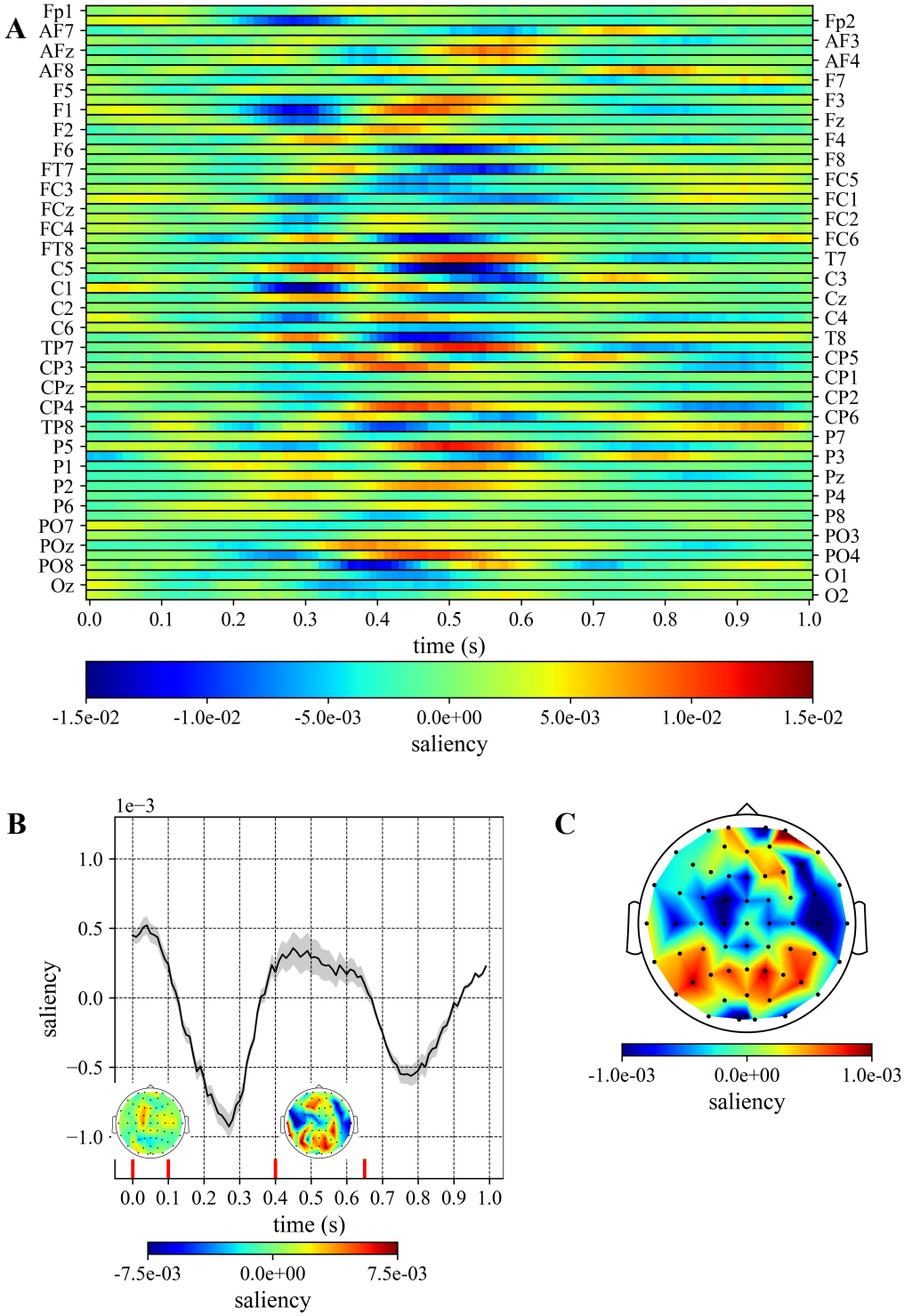


Figure 2.6 – Cross-subject saliency: target. (A) The 2D cross-subject saliency map is reported as a heatmap. (B) Temporal cross-subject saliency pattern, obtained by averaging the 2D saliency map across all electrodes. The time intervals where the saliency is higher are denoted by vertical red lines. For these specific intervals, the topological representation of the spatial cross-subject saliency pattern is also reported, obtained by averaging the saliency values of each electrode within each time interval. (C) Topological representation of the spatial cross-subject saliency map, obtained by averaging the saliency values of each electrode over the entire epoch (0-1000 ms).

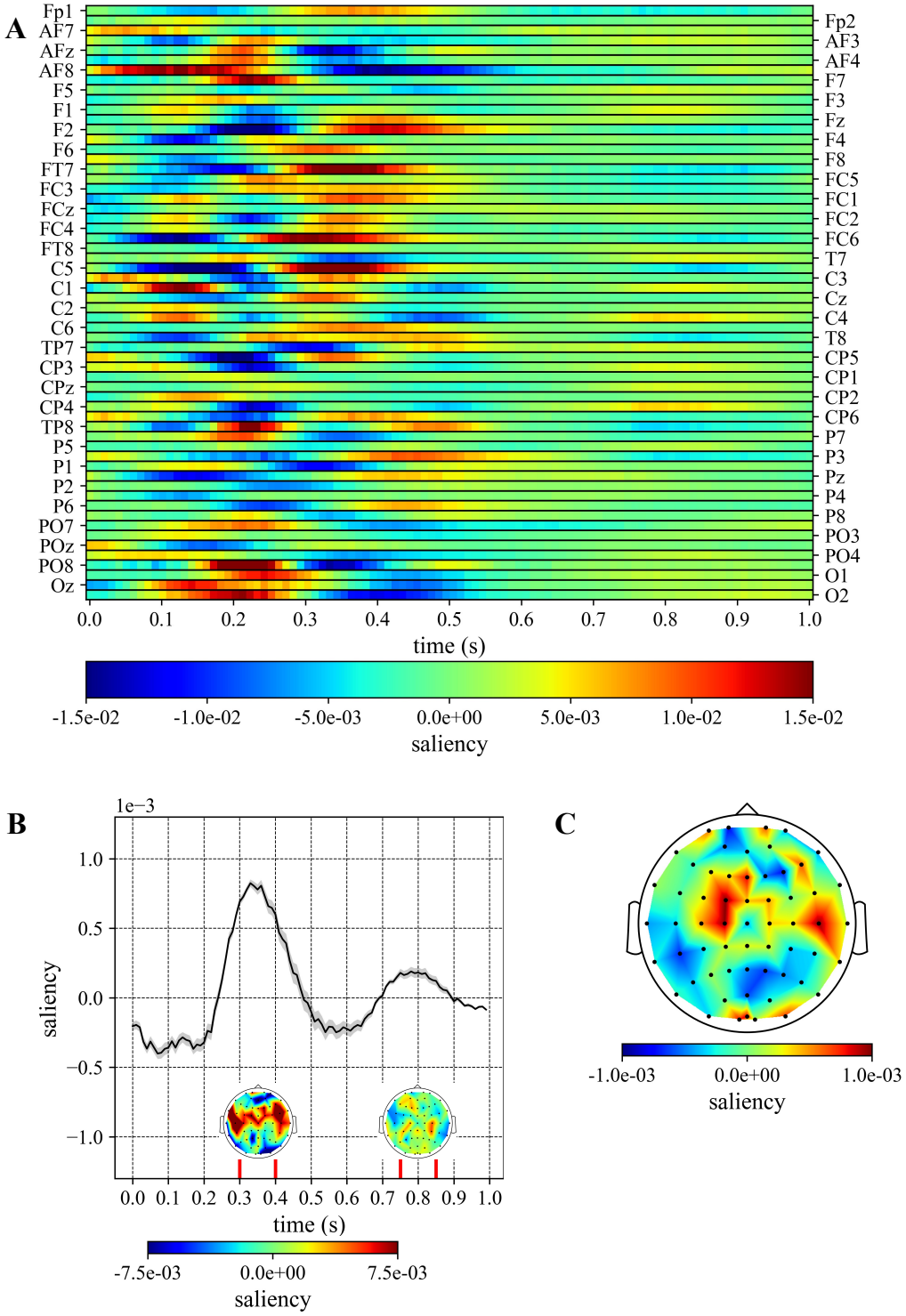


Figure 2.7 – Cross-subject saliency: distractor. (A) The 2D cross-subject saliency map is reported as a heatmap. (B) Temporal cross-subject saliency pattern, obtained by averaging the 2D saliency map across all electrodes. The time intervals where the saliency is higher are denoted by vertical red lines. For these specific intervals, the topological representation of the spatial cross-subject saliency pattern is also reported, obtained by averaging the saliency values of each electrode within each time interval. (C) Topological representation of the spatial cross-subject saliency map, obtained by averaging the saliency values of each electrode over the entire epoch (0-1000 ms).

Regarding the temporal cross-subject saliency patterns (Figures 2.6B and 2.7B), the temporal windows that mostly contributed to the discrimination were different in the two

conditions: they were 0-100 ms and 400-650 ms post-stimulus for the target condition (see intervals between vertical red lines in Figure 2.6B), and 300-400 ms and 750-850 ms post-stimulus for the distractor condition (see intervals between vertical red lines in Figure 2.7B). In addition, by visually inspecting the spatial patterns (Figures 2.6C and 2.7C) it is evident that the electrodes more class-discriminative over the entire epoch (0-1000 ms) were parietal sites (P1, P3, P5, P7, Pz, P2, P4, P5) for the target condition and sites from central to frontal areas (C1, C6, FC1, FCz, FC2, Fz) for the distractor condition, indeed these sites are characterized by intense red colour in the map. However, distinct relevant intervals (i.e., between the vertical red lines in Figures 2.6B and 2.7B) may be related to different electrode contributions, as reported in the spatial representations within the vertical red lines in Figures 2.6B and 2.7B. In particular, these showed a stronger involvement of sites from parieto-occipital to centro-parietal areas within 400-650 ms than 0-100 ms post-stimulus for the target condition. In addition, for the distractor condition, the spatial distribution related to the interval 300-400 ms post-stimulus highlighted a strong involvement of sites from central to frontal areas, while within 750-850 ms post-stimulus the more contributing electrodes lied also in backward sites (e.g., centro-parietal sites).

Cluster analysis: subject-specific and trial-specific saliency

In Figures 2.8 and 2.9 the temporal (left panels, thick black lines) and spatial (right panels) cluster-specific saliency patterns for each of the 4 clusters are reported for target and distractor conditions, respectively. Left panels contain also the temporal subject-specific saliency patterns (thin black lines) belonging to each cluster.

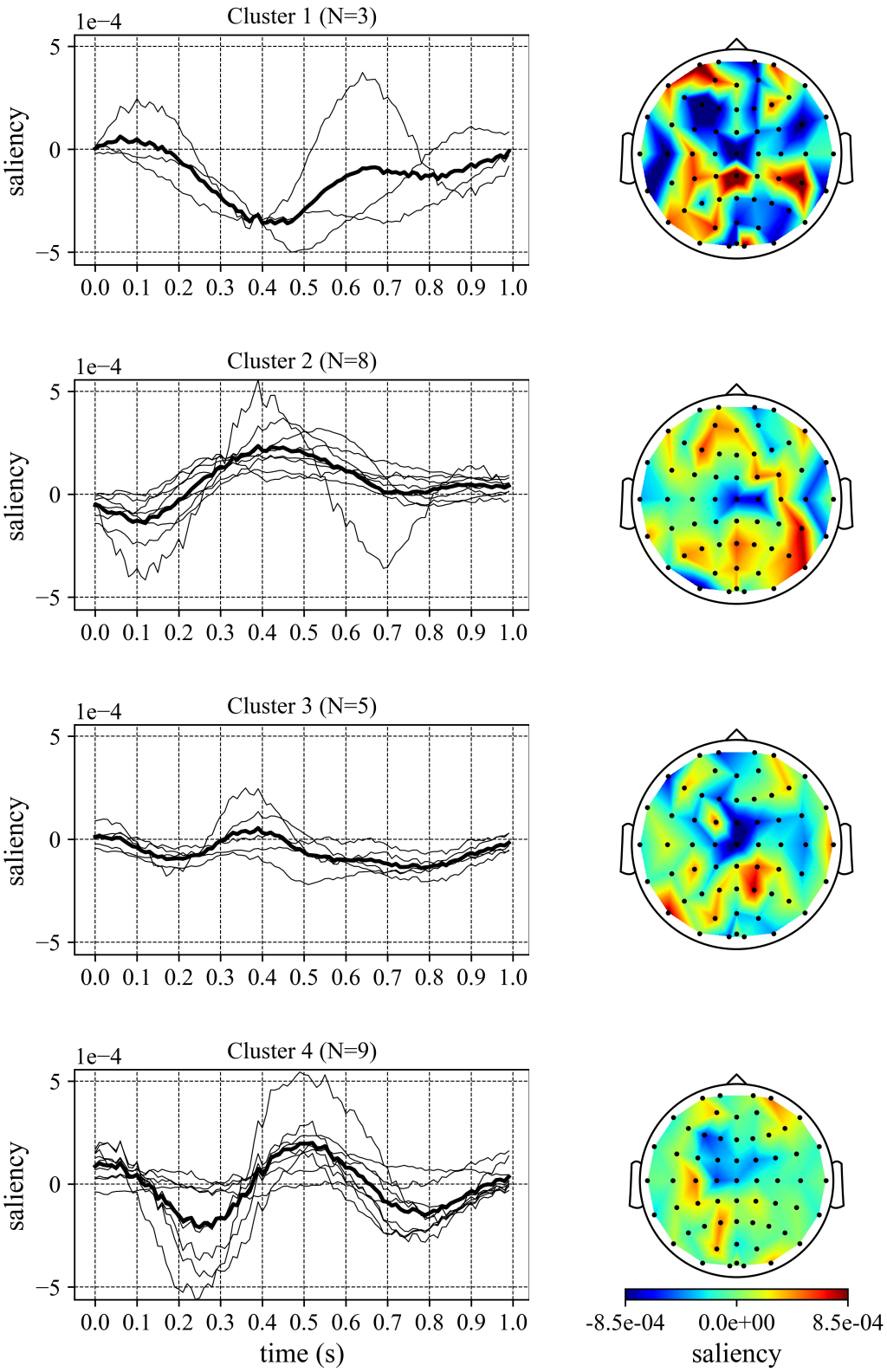


Figure 2.8 – Cluster analysis: target. For each cluster individuated by hierarchical agglomerative clustering, the left panel displays the temporal cluster-specific saliency pattern (thick line) together with the temporal subject-specific saliency patterns (thin lines) defining each cluster, while the topological map on the right displays the spatial cluster-specific saliency pattern.

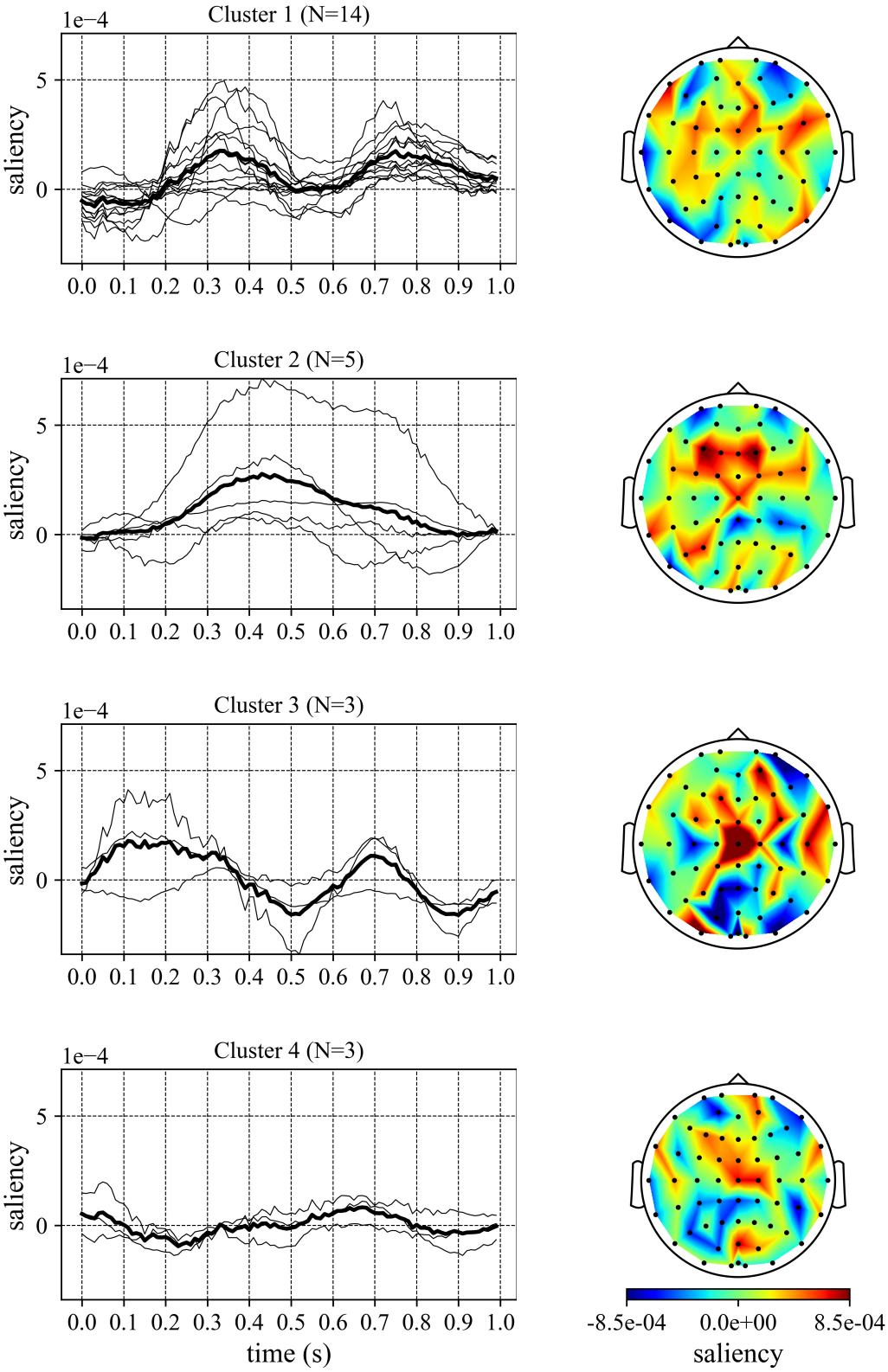


Figure 2.9 – Cluster analysis: distractor. For each cluster individuated by hierarchical agglomerative clustering, the left panel displays the temporal cluster-specific saliency pattern (thick line) together with the temporal subject-specific saliency patterns (thin lines) defining each cluster, while the topological map on the right displays the spatial cluster-specific saliency pattern.

For the target condition, most of the temporal subject-specific saliency patterns turned out to be grouped into two clusters: one (cluster 4 with $N = 9$ subjects) evidenced higher (positive)

saliency in a relatively narrow temporal window centered at around 500 ms, the other (cluster 2 with $N = 8$ subjects) exhibited higher saliency in a larger temporal window approximately around 450 ms. In only a few cases (cluster 3 with $N = 5$ subjects), an earlier time window (approximately between 250 ms and 450 ms) appeared more salient, although at lower levels compared to previous clusters. For these clusters, mainly centro-parietal and parietal electrodes were more discriminative, with a spatial distribution modulated depending on the specific cluster, i.e., more right-lateralized distribution for clusters 2, 3 and left-lateralized for cluster 4. Finally, cluster 1 (with only $N = 3$ subjects) seems to collect exceptions not falling in any of the previous clusters (clusters 2-4); the latter ones, although with clear differences between one cluster and the other, were characterized by a main positive peak (unimodal patterns) mostly developing before 500 ms, while patterns in cluster 1 did not exhibited such trait.

Conversely, for the distractor condition, bimodal distributions appeared evident (i.e., two main peaks can be individuated in the temporal patterns). Indeed, most of the temporal subject-specific saliency patterns exhibited two peaks within two temporal windows centered at around 350 ms and 750 ms (cluster 1 with $N = 14$ subjects). A few subjects (cluster 3 with $N = 3$) displayed a similar bimodal pattern but with the two peaks slightly anticipated. Only in a few cases, unimodal temporal patterns emerged with higher latencies, i.e., within windows centered at around 450 ms post-stimulus (cluster 2 with $N = 5$ subjects). For these clusters, mainly centro-frontal and frontal electrodes were more discriminative, with a different spatial distribution depending on the specific cluster, i.e., a more dispersed centro-frontal distribution for cluster 1, a more frontal distribution focused around the midline for cluster 2, and a more central distribution focused around Cz for cluster 3. Finally, cluster 4 (with only three subjects) is characterized by very small saliency values.

Lastly, for a single representative subject belonging to each previously computed cluster, Figures 2.10 and 2.11 display the EEG evoked potentials (Figure 2.10A and Figure 2.11A) and the temporal saliency patterns (Figure 2.10B and Figure 2.11B) for the target condition and distractor condition, respectively, at the level of single subject. Specifically, Figure 2.10A and Figure 2.11A report the average of EEG (target or distractor) trials for the specific subject (i.e., a subject-level EEG-derived representation) while Figure 2.10B and Figure 2.11B report the average of the saliency associated to the same trials of the same subject (i.e., the WS CNN+ET representation). In the same figures, the corresponding patterns at the level of single trials of the same subject (Figure 2.10C and Figure 2.11C reporting EEG over single trials, and Figure 2.10D and Figure 2.11D reporting CNN-derived saliency over single trials) are shown (see Section *Explanation technique: saliency maps computation and processing* for further details about the performed processing). It is worth noticing that in this case, at variance with Figures 2.8 and 2.9, the displayed quantities (both CNN-based saliency patterns and EEG patterns) refer only to a subset of electrodes (more parietal and more frontal in case of the target condition and distractor condition, respectively). It appears evident how saliency patterns (Figure 2.10B, D and Figure 2.11B, D) could enhance meaningful features not or only little evident in quantities directly derived from EEG, i.e., single-subject evoked potentials (Figure 2.10 A and 2.11A) and single EEG trials (Figure 2.10C and 2.11C), see also Section 2.4.

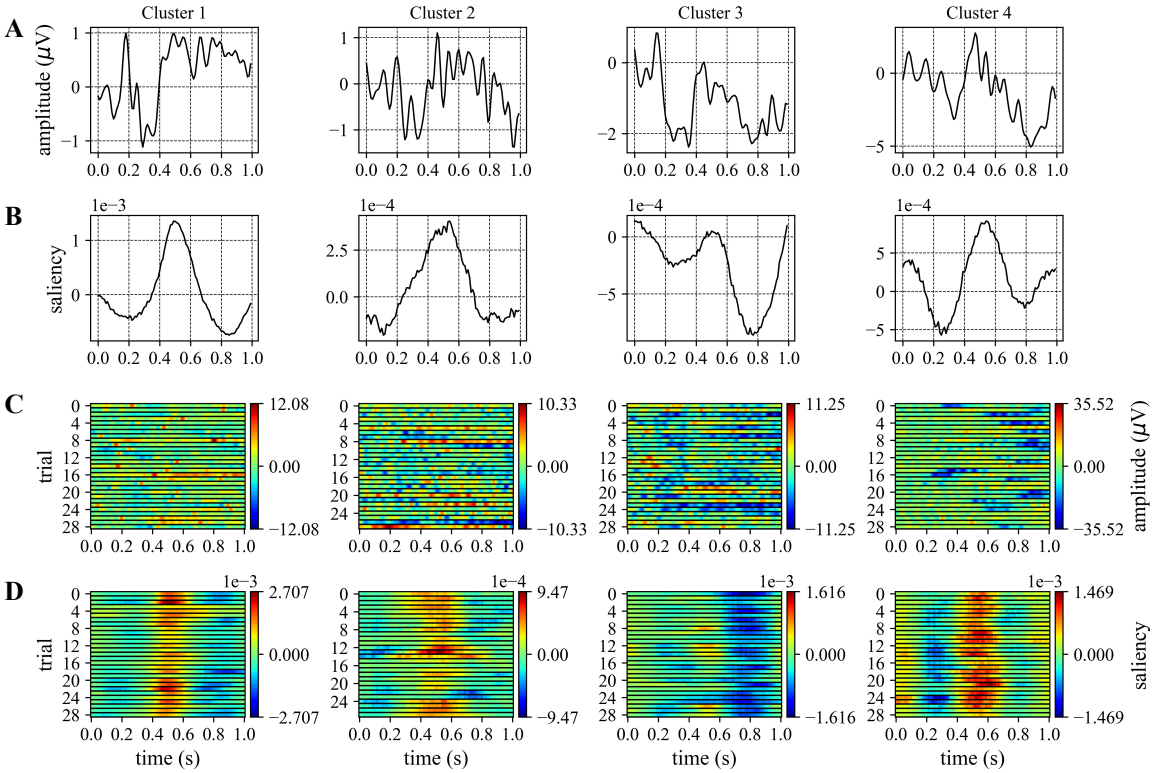


Figure 2.10 – Comparison between CNN-derived saliency and EEG-derived representations at single-subject and single-trial levels: target. These representations are displayed for one representative subject for each of the four clusters of Figure 2.8 (each column of Figure 2.10 is related to a specific cluster). (A, B) Evoked potentials directly derived from EEG trials (Figure 2.10A) and temporal WS CNN-derived saliency patterns (Figure 2.10B) at the level of single subject; both these representations involve averaging across trials of the same condition and across a subset of electrodes (P3, P1, Pz, P2, P4, PO3, POz, PO4). See Section *Explanation technique: saliency maps computation and processing* for details. (C, D) Single EEG trials (Figure 2.10C) and WS CNN-derived saliency pattern (Figure 2.10D) at the level of single trials: each row corresponds to a trial, and the representation in each row still involves averaging across the subset of electrodes. In practice, the patterns in Figure 2.10A and 2.10B correspond to averaging, across the rows, the representations in Figure 2.10C and 2.10D, respectively.

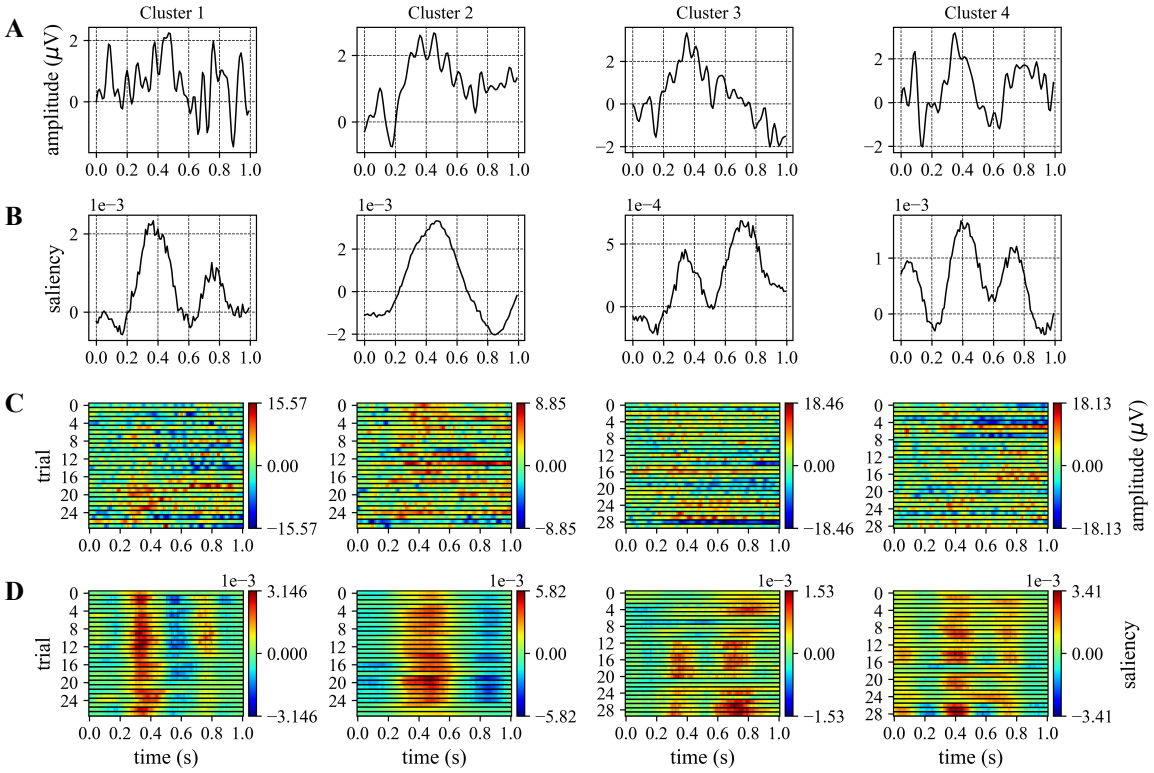


Figure 2.11 – Comparison between CNN-derived saliency and EEG-derived representations at single-subject and single-trial levels: distractor. These representations are displayed for one representative subject for each of the four clusters of Figure 2.9 (each column of Figure 2.11 is related to a specific cluster). (A, B) Evoked potentials directly derived from EEG trials (Figure 2.11A) and temporal WS CNN-derived saliency patterns (Figure 2.11B) at the level of single subject; both these representations involve averaging across trials of the same condition and across a subset of electrodes (FC1, FC2, C1, C2, CP1, CP2). See Section *Explanation technique: saliency maps computation and processing* for details. (C, D) Single EEG trials (Figure 2.11C) and WS CNN-derived saliency pattern (Figure 2.11D) at the level of single trials: each row corresponds to a trial, and the representation in each row still involves averaging across the subset of electrodes. In practice, the patterns in Figure 2.11A and 2.11B correspond to averaging, across the rows, the representations in Figure 2.11C and 2.11D, respectively.

2.4. DISCUSSION

In this study, the combination of a CNN (here EEGNet) with an ET (here gradient-based saliency maps) was adopted as a data-driven EEG analysis tool to investigate the electrophysiological signatures associated to P3 subcomponents (i.e., P3a and P3b), using EEG signals recorded during a 3-stimulus oddball paradigm. The adopted CNN+ET, by computing input saliency representations $g\left(X_i^{(s)}\right)$, allows a direct understanding of the more relevant spatial and temporal input samples when discriminating between standard, target and distractor stimuli. In addition, coupling the CNN+ET with a proper CNN training strategy, such as leave-one-subject-out strategy and within-subject strategy, the obtained relevance results more focused on features shared across subjects (i.e., common task-related features) and on subject-specific features, respectively. Therefore, depending on the training strategy, CNN+ET could provide useful information about the neural signatures belonging to the input domain more related to the specific task investigated or more related to the single subject. Furthermore, due to the nature of the supervised learning approached, performed to provide a discrimination between the contrasted conditions using single EEG trial as input, the provided CNN+ET can be used to investigate the more discriminative features already at the level of single trial. In the following, once commented the CNN performance in the addressed classification task, these aspects will be separately discussed.

2.4.1. CNN performance

EEGNet scored significantly higher performance when using cross-subject distributions as input during training (LOSO) than subject-specific input distributions (WS), especially in distinguishing standards vs. distractors (see Figure 2.5). This could be due to the extremely compact dataset used in this study, consisting of 188 trials on average per subject (the number of trials per subjects depended on the pre-processing procedure, see Section 2.1). Indeed, when training EEGNet with subject-specific distributions, only 170 training trials on average were used within each fold. Conversely, during LOSO trainings 24 subjects' signals were exploited, leading to 4512 training trials on average. Therefore, despite LOSO trainings were inherently more challenging due to the subject-to-subject variability in the input distributions, EEGNet performance resulted higher than WS trainings possibly due to the availability of a larger training set. In addition, it is worth mentioning that the LOSO performance achieved by EEGNet in the addressed 3-classes decoding problem resulted similar to that obtained in more common 2-classes P300 decoding problems (i.e., target vs. standard conditions) [42], despite the increased difficulty in the classification task due to the discrimination between more than two conditions (i.e., standard vs. target vs. distractor).

Furthermore, EEGNet performance could have been affected by the specific applied pre-processing. In this study, we kept the pre-processing pipeline unchanged respect to the study by Cavanagh et al [48] where the adopted dataset was collected and presented (see Section 2.1). However, this filtering may limit the capability of the network to autonomously identify the bands most relevant for classification. Therefore, we tested also the CNNs performance when changing the band-pass filtering from 0.1-20 Hz to 0.1-40 Hz in the pre-processing pipeline; in this case the CNN could leverage additional information to solve the decoding task or, conversely, choose to filter out unrelated information. Providing a larger frequency content

in input, a moderate but significant improvement in CNN performance was obtained; in the LOSO strategy, av-AUROCs improved to 77.7±1.1% compared to 76.2±1.3% ($p = 1.87 \cdot 10^{-2}$, Wilcoxon signed-rank test) and in the WS strategy av-AUROCs improved to 73.7±1.4% compared to 70.5±1.2% ($p = 1.29 \cdot 10^{-3}$).

2.4.2. CNN-based cross-subject analysis

When compared to ERPs (Figures 2.3 and 2.4, obtained according to the canonical grand average over all EEG trials and subjects), the temporal cross-subject saliency patterns reported in Figures 2.6 and 2.7 matched the P3b and P3a timings – 400-650 ms and 350-400 ms post-stimulus, respectively for target stimuli and distractor stimuli (see Figures 2.3B, 2.4B, 2.6B, 2.7B). Similarly, the spatial cross-subject saliency patterns matched the P3b and P3a scalp distributions, both when the spatial saliency patterns were computed over all time samples (Figures 2.3C, 2.4C, 2.6C, 2.7C) and within 400-650 ms and 350-400 ms post-stimulus (see Figures 2.6B, 2.7B), respectively for the target condition and distractor condition. It is worth noticing that this comparison was made possible as cross-subject saliency patterns were computed performing a grand average by construction (see Section *Explanation technique: saliency maps computation and processing*), as done to obtain ERPs. From this analysis emerges the first of the main contributions of the proposed approach. Indeed, the findings suggest that the CNN, without any a priori knowledge about the neural signatures related to target and distractor stimuli, during the supervised learning was able to automatically capture meaningful class-discriminative features related to P3b and P3a. In addition, the CNN+ET combination evidenced temporal and spatial patterns that were shared across subjects (i.e., being robust across subjects), resulting from a common strategy exploited by the learning system to distinguish between the three output classes using multiple subjects' signals (see Section *Training strategy*).

Moreover, due the supervised task addressed with the CNN – i.e., discrimination from single EEG trials between standard, target and distractor stimuli – the CNN+ET was able not only to evidence correlates related to P3a and P3b, but potentially also those related to other P3 subcomponents. Indeed, other ERPs can be elicited by distractor stimuli, such as the novelty P300 (which is a third and later subcomponent after the P3b) [12,57]; together with P3a, these components appear to be variants of the same ERP, varying on the basis of attentional and task demands. In particular, the late relevant window (750-850 ms) in the response to distractor stimuli (Figure 2.7), could be related to a later component such as the novelty P300.

2.4.3. CNN-based single-subject and single-trial analysis

The cluster analysis performed on temporal subject-specific saliency patterns evidenced distinct clusters at subject-level in the temporal and spatial domains that deviated from the shared pattern across subjects discussed in the previous section (see Figures 2.8, 2.9B,C vs. Figures 2.6, 2.7B,C). Therefore, it is possible to better analyze the subject-to-subject variability, by defining clusters of subjects that responded similarly to stimuli, and differently from other subjects. In particular, temporal subject-specific saliency patterns related to target stimuli exhibited two more frequent strategies (see clusters 4, 2 in Figure 2.8). These, although presenting a single positive peak and mainly involving parietal electrodes as in the

corresponding general cross-subject patterns (Figure 2.6B,C), revealed specific and distinguishable traits as they are centered at two slightly different time points (i.e., 500 ms and 450 ms post-stimulus) with a different dispersion across time points (i.e., cluster 2 resulted more dispersed in time) and with a different lateralization of the more contributing electrodes. When looking to patterns related to distractor stimuli in the most frequent strategy (see cluster 1 in Figure 2.9, including more than half of the subjects), these patterns appeared similar to the corresponding general cross-subject patterns (in agreement with a large proportion of subjects inside the cluster), while patterns in the other clusters exhibited larger deviation from the general cross-subject patterns (e.g., see cluster 2 in Figure 2.9, where temporal patterns with a single peak occurred as opposed to bimodal patterns). Finally, the cluster analysis evidenced some less reliable clusters that are less populated (e.g., cluster 1 in Figure 2.8 which seems to collect exceptions rather than representing a real cluster); this may be the consequence of the small subject-specific datasets. In case of larger datasets, a larger number of trials per participant my favor the identification of meaningful features in one subject similar as in other subjects, avoiding the occurrence of cluster seemingly collecting exceptions (this may be tested in future studies on larger datasets).

Importantly, the temporal subject-specific saliency patterns (left panels in Figures 2.8 and 2.9) showed relevant temporal samples potentially related to the P3b and P3a already at the level of single subject. In particular, the potential enhancement of these P3 components in saliency representations becomes clearer especially when comparing them with evoked potentials at the level of single subject. Indeed, from Figures 2.10 and 2.11, temporal saliency patterns at the level of single subject enhanced the relevant processes underlying the task in all reported cases compared to the evoked potentials counterpart, where meaningful neural signatures were less clear and distinguishable (e.g., see representative patterns in Figure 2.10A for clusters 1, 2 or in Figure 2.11A for cluster 1). However, it is worth mentioning that in some examples the correlate was noticeable also in evoked potentials at the level of single subject (e.g., single-subject representations in Figures 2.10A and 2.11A for cluster 4), but saliency representations resulted smoother and sharper. These considerations about saliency patterns become even more relevant at the level of single trial, where the saliency patterns seemed to preserve the well-defined temporal structure across different trials. On the contrary, single EEG trials resulted highly de-structured in time, with only few of them exhibiting P3b- and P3a-related correlates (e.g., trial 19 for the representative subject of cluster 3 of Figure 2.10C, trials 19, 20 for the representative subject of cluster 1 of Figure 2.11C), but without any clear coherence across recording trials, overall. From these results, the second main contribution of this study emerges, consisting in disclosing the potentialities of the CNN+ET combination to enhance the correlates related to the main P3 subcomponents (here in healthy controls) already at the single-trial level (and scaling up, at the single-subject level); the proposed method demonstrates the ability to empower the analysis of P3 modulations at the single-trial and single-subject levels, overcoming the main limitations of a canonical analysis based on evoked potentials (i.e., grand average across EEG trials and subjects). In particular, this is obtained by formalizing a processing CNN-based pipeline, that allows the analysis to be performed at multiple scales and domains (across-subjects, within-subject, within-trial, in time-space domain, or separately in the temporal and spatial domain). In prospective, the proposed data-driven analysis tool based on a CNN could be used to advance the investigation at single-

subject level (e.g., to assess between-subject variability) and also at single-trial level (e.g., to assess within-subject variability by analyzing differences between early vs. late recorded trials or correct/incorrect response trials), both in healthy subjects but also in patients with neurological or psychiatric disorders involving P3 alterations, e.g., Parkinson's disease or schizophrenia. In particular, enhancing neural signatures of stimuli processing at single subject scale and at single trial scale is of great relevance to explore the functional relationships of these neural features with human performance, and to boost the comprehension of the neural processes linking sensory stimulation, cognition and behavior.

It is worth remark that, despite deep learning-based decoders are known to require large datasets during training, the adoption of carefully designed solutions (in terms of number of parameters to fit) such as EEGNet-derived algorithms, can be used to derive useful representations in a CNN+ET framework even using small datasets, e.g., comprising less than 200 trials per subject in the addressed 3-stimulus oddball paradigm, as suggested by our results. However, the low number of trials for each subject of the adopted dataset may have affected the representations at the single-subject level, obtained with the WS strategy, and, thus, the performed analysis should be extended on larger datasets (comprising more trials per subject), to produce a more robust validation of single-subject representations.

2.5. CONCLUSIONS

In conclusion, we investigated the P3 in its main subcomponents with a CNN+ET workflow, analyzing in a data-driven way the more important spatial and temporal samples of EEG signals in healthy controls during a 3-stimulus oddball paradigm. The composition CNN+ET, depending on the CNN training strategy (cross-subject and within-subject), was able to extract EEG neural signatures not only shared across subject (i.e., robust task-related features) as the ones obtained with a canonical ERP analysis, but also specific for each subject and for each trial, both in the temporal and spatial domains. Therefore, the CNN+ET can be seen as a transformation of EEG signals able to enhance EEG neural signatures already at the level of single trial (and scaling up at the level of single subject), providing information that could increase the understanding of the neural processes underlying the relationship between incoming sensory stimuli, cognition and behavior.

Future developments may involve the inclusion in the workflow of elements aimed to further improve the comprehension of the learned CNN features (i.e., adoption of directly interpretable layers, not requiring post-hoc interpretation techniques) [35,58], the adoption of larger datasets, and the application of this approach to signals recorded from patients with psychiatric disorders, for a better characterization of the neural signatures associated to these disorders and of their relationships with clinical signs, potentially contributing to characterize novel biomarkers for diagnosis and monitoring.

2.6. SUPPLEMENTARY MATERIALS

2.6.1. EEGNet hyper-parameter details

To describe the CNN in Supplementary Table 2.1 we will refer to the hyper-parameters of the involved layers. Each convolutional layer is characterized by the number of convolutional kernels (K), kernel size (F), stride size (S) and padding size (P). In addition, depthwise convolution introduced also a depth multiplier (D) specifying the number of kernels to learn for each input feature map. Hyper-parameters will be denoted by a superscript and a subscript. The superscript indicates the specific block the layer belongs to, using the acronyms "ST", "T", "FC". The subscript indicates which convolutional layer inside the block the hyper-parameters refer to (starting from 0). Lastly, pooling layers were described by pool size (F_p) and stride (S_p), with the corresponding superscript. Both convolutions and poolings were 2D; thus, F, S, P, F_p, S_p were tuples of two integers: the first referred to the spatial dimension, while the second to the temporal dimension. Lastly, the temporal dimension changed across pooling operators and was denoted with T_p. Regarding the fully-connected layer in the last block, the number of neurons was denoted with N^{FC} and was equal to the number of classes to decode ($N_c = 3$).

Supplementary Table 2.1 – Architecture details of the used EEGNet adaptation. Each layer is provided with its name, main hyper-parameters and number of trainable parameters. The total number of trainable parameters was 1259. In all layers, where not specified, stride (S) and padding (P) were set to $(1,1)$ and $(0,0)$, respectively. The number of channels and time samples provided as input to the CNN were $C = 60$ and $T = 100$, see Section 2.3 in the manuscript. The temporal dimension changed from T to $T//18$ along the entire CNN (where the symbol//denotes the floor division) due to average poolings.

Block	Layer name	Hyper-parameters	Number of trainable parameters
Spatio-temporal (ST)	Input	$K_0 = 1$	0
	Conv2D	${K_0}^{\mathrm{ST}} = 8$, ${F_0}^{\mathrm{ST}} = (1,51)$, ${P_0}^{\mathrm{ST}} = (0,25)$	${F_0}^{\mathrm{ST}}[0] \cdot {F_0}^{\mathrm{ST}}[1] \cdot {K_0}^{\mathrm{ST}} \cdot K_0$
	BatchNorm2D	$momentum = 0.99$	$2 \cdot {K_0}^{\mathrm{ST}}$
	Depthwise-Conv2D	${D_1}^{\mathrm{ST}} = 1$, ${K_1}^{\mathrm{ST}} = {K_0}^{\mathrm{ST}} \cdot {D_1}^{\mathrm{ST}}$, ${F_1}^{\mathrm{ST}} = (C, 1)$	${F_1}^{\mathrm{ST}}[0] \cdot {F_1}^{\mathrm{ST}}[1] \cdot {K_1}^{\mathrm{ST}}$
	BatchNorm2D	$momentum = 0.99$	$2 \cdot {K_1}^{\mathrm{ST}}$
	ELU		0
	AvgPool2D	${F_p}^{\mathrm{ST}} = {S_p}^{\mathrm{ST}} = (1,3) \rightarrow {T_p}^{\mathrm{ST}} = T//3$	0
	Dropout	$p = 0.5$	0
Temporal (T)	Separable-Conv2D	${D_0}^{T} = 1, {K_0}^{T} = {K_1}^{ST} \cdot {D_0}^{T}$, ${F_0}^{T} = (1,17)$, ${P_0}^{T} = (0,8)$	${F_0}^{T}[0] \cdot {F_0}^{T}[1] \cdot {K_0}^{T} + \left({K_0}^{T}\right)^2$
	BatchNorm2D	$momentum = 0.99$	$2 \cdot {K_0}^{T}$
	ELU		0
	AvgPool2D	${F_p}^{T} = {S_p}^{T} = (1,6) \rightarrow {T_p}^{\mathrm{ST}} = T//18$	0
	Dropout	$p = 0.5$	0

Fully-connected (FC)	Flatten		0
	Fully-Connected	$N^{FC} = 3$	$N^{FC} \cdot ({T_p}^T \cdot {K_0}^T + 1)$
	Softmax		0

The adopted EEGNet was lightly different from the original formulation [33]. In particular, since EEGNet was designed to accept EEG signals sampled at 128 Hz as input, we scaled down the temporal pool sizes and kernel lengths accordingly. In addition, due to the compact size of the used dataset, we further reduced the number of trainable parameters to fit by using a unitary depth multiplier in all depthwise convolutions (both in the spatio-temporal block and in the temporal block). Lastly, we did not exploit kernel max norm constraint in the architecture as it resulted detrimental for the performance in previous analyses on the same dataset.

2.6.2. Hierarchical agglomerative clustering details

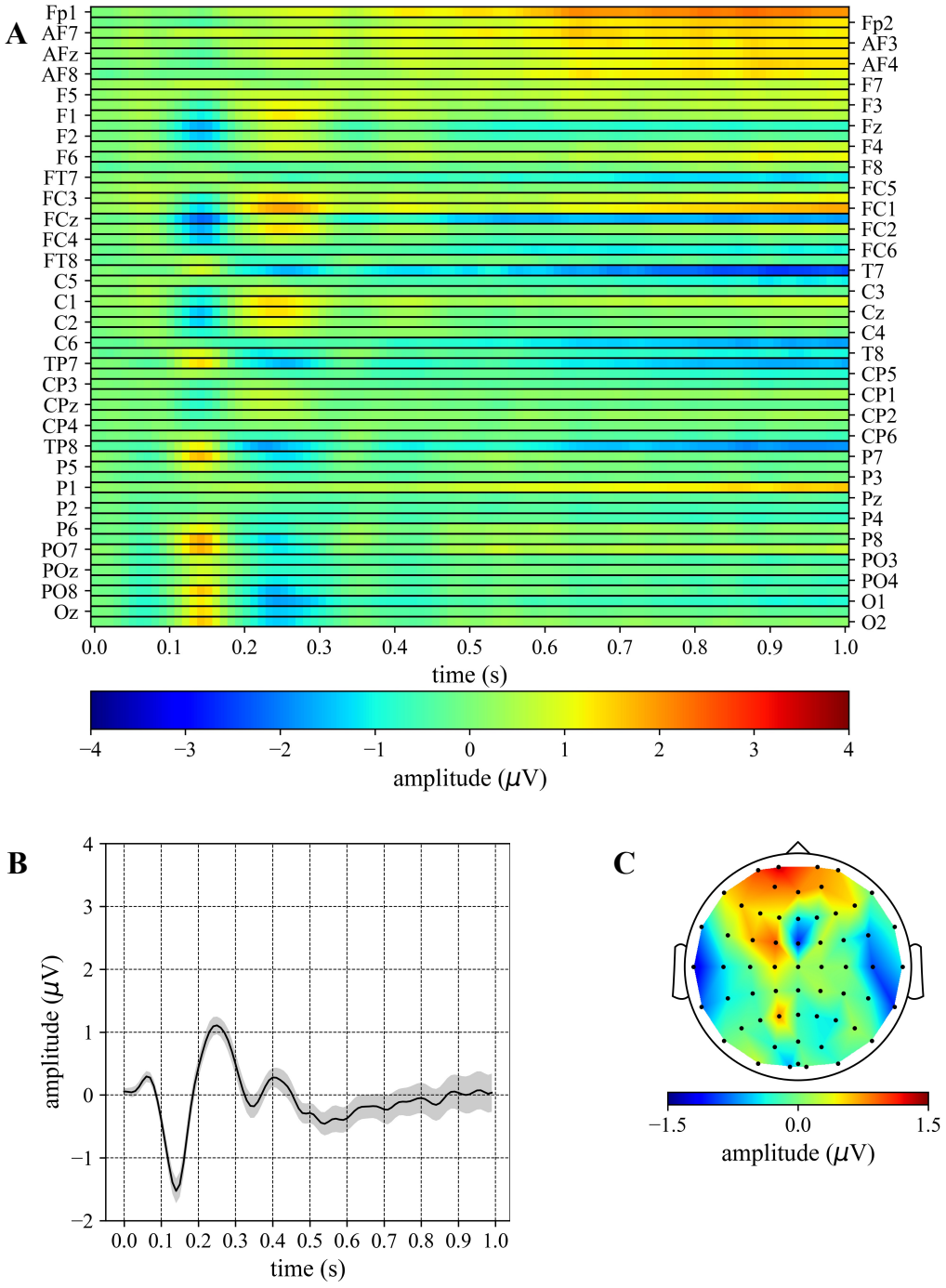
In the following, the distance metric adopted to apply the hierarchical agglomerative clustering (HAC) is described. During HAC, pairwise distances between observations in T-dimensional space were computed, indicating with T the number of time samples (as in the manuscript). The two observations corresponded to two temporal saliency patterns belonging to two different subjects (i.e., subject-specific temporal saliency patterns). The correlation between $u, v \in \mathbb{R}^{\mathrm{T}}$ was computed to define a correlation distance metric $d(u, v)$ as:

$$d(u, v) = 1 - \frac{(u-\bar{u})\cdot(v-\bar{v})}{\|(u-\bar{u})\|_2 \|(v-\bar{v})\|_2},$$

where $\bar{u}$, $\bar{v}$ are the mean values of the arrays u and v, while $\cdot$ denotes the dot product.

Supplementary Table 2.2 – F1 scores and AUPRs for the target and distractor conditions while training CNNs with the LOSO and WS training strategies (mean value ± standard error of the mean). It is worth noticing that these results are similar or slightly lower (for the WS strategy) than state-of-the-art results reported at the level of single trial in binary P3 classification tasks (discriminating only standard vs. target) [43], despite here a more challenging, three-class classification problem is addressed.

Training strategy	Class	F1 score (%)	AUPR (%)
LOSO	Target	37.3 ± 2.3	38.0 ± 2.7
	Distractor	48.5 ± 1.8	51.0 ± 2.1
WS	Target	35.2 ± 1.8	45.1 ± 2.1
	Distractor	34.7 ± 1.8	47.1 ± 2.2



Supplementary Figure 2.1 – ERP of the standard condition. (A) The grand average is reported as a 2D heatmap with electrodes and time steps along rows and columns, respectively. (B) The average temporal pattern obtained by averaging the 2D heatmap of panel A across the subset of electrodes: FC1, FC2, C1, C2, CP1, CP2. The shaded area represents the mean value ± standard error of the mean, while the thick line represents the mean value. (C) Topological representation of the average contribution of each electrode across all time samples of the 2D heatmap.

2.7. REFERENCES

[1] Schröder E, Kajosch H, Verbanck P, Kornreich C and Campanella S 2016 Methodological Considerations about the Use of Bimodal Oddball P300 in Psychiatry: Topography and Reference Effect *Front. Psychol.* **7**

[2] Davies P L 2010 Middle and late latency ERP components discriminate between adults, typical children, and children with sensory processing disorders *Front. Integr. Neurosci.* **4**

[3] Boutros N N, Gjini K and Arfken C L 2011 Advances in electrophysiology in the diagnosis of behavioral disorders *Expert Opinion on Medical Diagnostics* **5** 441–52

[4] Polich J and Herbst K L 2000 P300 as a clinical assay: rationale, evaluation, and findings *International Journal of Psychophysiology* **38** 3–19

[5] Jeon Y-W and Polich J 2003 Meta-analysis of P300 and schizophrenia: Patients, paradigms, and practical implications *Psychophysiology* **40** 684–701

[6] Roth W T, Pfefferbaum A, Kelly A F, Berger P A and Kopell B S 1981 Auditory event-related potentials in schizophrenia and depression *Psychiatry research* **4** 199–212

[7] Wada M, Kurose S, Miyazaki T, Nakajima S, Masuda F, Mimura Y, Nishida H, Ogyu K, Tsugawa S, Mashima Y, Plitman E, Chakravarty M M, Mimura M and Noda Y 2019 The P300 event-related potential in bipolar disorder: A systematic review and meta-analysis *Journal of Affective Disorders* **256** 234–49

[8] Cui T, Wang P P, Liu S and Zhang X 2017 P300 amplitude and latency in autism spectrum disorder: a meta-analysis *European child & adolescent psychiatry* **26** 177–90

[9] Paszkiel S 2020 Data Acquisition Methods for Human Brain Activity *Analysis and Classification of EEG Signals for Brain–Computer Interfaces* Studies in Computational Intelligence vol 852 (Cham: Springer International Publishing) pp 3–9

[10] Sutton S, Braren M, Zubin J and John E R 1965 Evoked-Potential Correlates of Stimulus Uncertainty *Science* **150** 1187–8

[11] Farwell L A and Donchin E 1988 Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials *Electroencephalography and Clinical Neurophysiology* **70** 510–23

[12] Polich J 2007 Updating P300: An integrative theory of P3a and P3b *Clinical Neurophysiology* **118** 2128–48

[13] Donchin E and Coles M G H 1988 Is the P300 component a manifestation of context updating? *Behav Brain Sci* **11** 357

[14] Ritter W and Vaughan H G J 1969 Averaged evoked responses in vigilance and discrimination: a reassessment. *Science* **164** 326–8

[15] Vaughan H G and Ritter W 1970 The sources of auditory evoked responses recorded from the human scalp *Electroencephalography and Clinical Neurophysiology* **28** 360–7

[16] Knight R T 1996 Contribution of human hippocampal region to novelty detection *Nature* **383** 256–9

[17] Wronka E, Kaiser J and Coenen A M L 2012 Neural generators of the auditory evoked potential components P3a and P3b. *Acta Neurobiol Exp (Wars)* **72** 51–64

[18] Näätänen R 1990 The role of attention in auditory information processing as revealed by event-related potentials and other brain measures of cognitive function *Behav Brain Sci* **13** 201–33

[19] Solís-Vivanco R, Rodríguez-Violante M, Rodríguez-Agudelo Y, Schilmann A, Rodríguez-Ortiz U and Ricardo-Garcell J 2015 The P3a wave: A reliable neurophysiological measure of Parkinson's disease duration and severity *Clin Neurophysiol* **126** 2142–9

[20] Bruder G E, Kroppmann C J, Kayser J, Stewart J W, McGrath P J and Tenke C E 2009 Reduced brain responses to novel sounds in depression: P3 findings in a novelty oddball task *Psychiatry Res* **170** 218–23

[21] Hada M, Porjesz B, Begleiter H and Polich J 2000 Auditory P3a assessment of male alcoholics *Biol Psychiatry* **48** 276–86
[22] Atkinson R J, Michie P T and Schall U 2012 Duration mismatch negativity and P3a in first-episode psychosis and individuals at ultra-high risk of psychosis *Biol Psychiatry* **71** 98–104
[23] Gaspar C M, Rousselet G A and Pernet C R 2011 Reliability of ERP and single-trial analyses *Neuroimage* **58** 620–9
[24] Rousselet G A and Pernet C R 2011 Quantifying the Time Course of Visual Object Processing Using ERPs: It's Time to Up the Game *Front Psychol* **2** 107
[25] Bridwell D A, Cavanagh J F, Collins A G E, Nunez M D, Srinivasan R, Stober S and Calhoun V D 2018 Moving Beyond ERP Components: A Selective Review of Approaches to Integrate EEG and Behavior *Front Hum Neurosci* **12** 106
[26] LeCun Y, Bengio Y and Hinton G 2015 Deep learning *nature* **521** 436
[27] Craik A, He Y and Contreras-Vidal J L 2019 Deep learning for electroencephalogram (EEG) classification tasks: a review *J. Neural Eng.* **16** 031001
[28] Roy Y, Banville H, Albuquerque I, Gramfort A, Falk T H and Faubert J 2019 Deep learning-based electroencephalography analysis: a systematic review *Journal of Neural Engineering* **16** 051001
[29] Lindsay G 2020 Convolutional Neural Networks as a Model of the Visual System: Past, Present, and Future *Journal of Cognitive Neuroscience* 1–15
[30] Lotte F, Bougrain L, Cichocki A, Clerc M, Congedo M, Rakotomamonjy A and Yger F 2018 A review of classification algorithms for EEG-based brain–computer interfaces: a 10 year update *Journal of Neural Engineering* **15** 031005
[31] Simões M, Borra D, Santamaría-Vázquez E, GBT-UPM, Bittencourt-Villalpando M, Krzemiński D, Miladinović A, Neural_Engineering_Group, Schmid T, Zhao H, Amaral C, Direito B, Henriques J, Carvalho P and Castelo-Branco M 2020 BCIAUT-P300: A Multi-Session and Multi-Subject Benchmark Dataset on Autism for P300-Based Brain-Computer-Interfaces *Front. Neurosci.* **14** 568104
[32] Faust O, Hagiwara Y, Hong T J, Lih O S and Acharya U R 2018 Deep learning for healthcare applications based on physiological signals: A review *Computer Methods and Programs in Biomedicine* **161** 1–13
[33] Lawhern V J, Solon A J, Waytowich N R, Gordon S M, Hung C P and Lance B J 2018 EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces *Journal of Neural Engineering* **15** 056013
[34] Schirrmeister R T, Springenberg J T, Fiederer L D J, Glasstetter M, Eggensperger K, Tangermann M, Hutter F, Burgard W and Ball T 2017 Deep learning with convolutional neural networks for EEG decoding and visualization *Human brain mapping* **38** 5391–420
[35] Borra D, Fantozzi S and Magosso E 2020 Interpretable and lightweight convolutional neural network for EEG decoding: Application to movement execution and imagination *Neural Networks* **129** 55–74
[36] Borra D, Fantozzi S and Magosso E 2020 EEG Motor Execution Decoding via Interpretable Sinc-Convolutional Neural Networks *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1113–22
[37] Paszkiel S and Dobrakowski P 2021 The Use of Multilayer ConvNets for the Purposes of Motor Imagery Classification *Automation 2021: Recent Achievements in Automation, Robotics and Measurement Techniques* Advances in Intelligent Systems and Computing vol 1390, ed R Szewczyk, C Zieliński and M Kaliczyńska (Cham: Springer International Publishing) pp 10–9

[38] Li J, Zhang Z and He H 2018 Hierarchical Convolutional Neural Networks for EEG-Based Emotion Recognition *Cogn Comput* **10** 368–80
[39] Liu J, Wu G, Luo Y, Qiu S, Yang S, Li W and Bi Y 2020 EEG-Based Emotion Classification Using a Deep Neural Network and Sparse Autoencoder *Front. Syst. Neurosci.* **14** 43
[40] Acharya U R, Oh S L, Hagiwara Y, Tan J H and Adeli H 2018 Deep convolutional neural network for the automated detection and diagnosis of seizure using EEG signals *Computers in biology and medicine* **100** 270–8
[41] Borra D, Fantozzi S and Magosso E 2020 Convolutional Neural Network for a P300 Brain-Computer Interface to Improve Social Attention in Autistic Spectrum Disorder *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1837–43
[42] Borra D, Fantozzi S and Magosso E 2021 A Lightweight Multi-Scale Convolutional Neural Network for P300 Decoding: Analysis of Training Strategies and Uncovering of Network Decision *Frontiers in Human Neuroscience*
[43] Liu M, Wu W, Gu Z, Yu Z, Qi F and Li Y 2018 Deep learning based on Batch Normalization for P300 signal detection *Neurocomputing* **275** 288–97
[44] Manor R and Geva A B 2015 Convolutional Neural Network for Multi-Category Rapid Serial Visual Presentation BCI *Frontiers in Computational Neuroscience* **9** 146
[45] Shan H, Liu Y and Stefanov T 2018 A Simple Convolutional Neural Network for Accurate P300 Detection and Character Spelling in Brain Computer Interface *Proceedings of the 27th International Joint Conference on Artificial Intelligence* IJCAI'18 (Stockholm, Sweden: AAAI Press) pp 1604–10
[46] Montavon G, Samek W and Müller K-R 2018 Methods for interpreting and understanding deep neural networks *Digital Signal Processing* **73** 1–15
[47] Simonyan K, Vedaldi A and Zisserman A 2014 Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps *arXiv:1312.6034 [cs]*
[48] Cavanagh J F, Kumar P, Mueller A A, Richardson S P and Mueen A 2018 Diminished EEG habituation to novel events effectively classifies Parkinson's patients *Clinical Neurophysiology* **129** 409–18
[49] Bradley M M and Lang P J 1999 International Affective Digitized Sounds (IADS-1): Stimuli, instruction manual, and affective ratings *Technical Report No. B-2* **Gainesville, FL: University of Florida, Center for Research in Psychophysiology**
[50] Delorme A and Makeig S 2004 EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis *Journal of Neuroscience Methods* **134** 9–21
[51] Nolan H, Whelan R and Reilly R B 2010 FASTER: Fully Automated Statistical Thresholding for EEG artifact Rejection *Journal of Neuroscience Methods* **192** 152–62
[52] Paszke A, Gross S, Chintala S, Chanan G, Yang E, DeVito Z, Lin Z, Desmaison A, Antiga L and Lerer A 2017 Automatic differentiation in PyTorch *NIPS-W*
[53] Kingma D P and Ba J 2017 Adam: A Method for Stochastic Optimization *arXiv:1412.6980 [cs]*
[54] Hand D J and Till R J 2001 A Simple Generalisation of the Area Under the ROC Curve for Multiple Class Classification Problems *Machine Learning* **45** 171–86
[55] Benjamini Y and Hochberg Y 1995 Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing *Journal of the Royal Statistical Society. Series B (Methodological)* **57** 289–300
[56] Müllner D 2011 Modern hierarchical, agglomerative clustering algorithms *arXiv:1109.2378 [cs, stat]*

[57] Barry R J, Steiner G Z, De Blasio F M, Fogarty J S, Karamacoska D and MacDonald B 2020 Components in the P300: Don't forget the Novelty P3! *Psychophysiology* **57** e13371
[58] Zhao D, Tang F, Si B and Feng X 2019 Learning joint space–time–frequency features for EEG decoding on small labeled data *Neural Networks* **114** 67–77

CHAPTER 3: DESIGN OF AN INTERPRETABLE CNN FOR P300 DECODING AND ANALYSIS IN THE FREQUENCY AND SPATIAL DOMAINS

The study reported in this chapter refers to the submitted journal paper entitled "A Bayesian-optimized design for an interpretable convolutional neural network to decode and analyze the P300 response in autism" D. Borra, E. Magosso, M. Castelo-Branco and M. Simões. Submitted to *Journal of Neural Engineering*. In this chapter, a novel interpretable CNN was proposed for P300 decoding, by increasing the interpretability of the CNN used in the previous chapters at the level of the first temporal convolutional layer. The design (in terms of the hyper-parameters) of the interpretable CNN was optimized using an automatic hyper-parameter search and was evaluated in multiple training strategies. Lastly, an intermediate explanation technique was developed to study P300 neural signatures in the frequency and spatial domains.

Event-Related Potentials (ERP), such as P300, can be analyzed in autism spectrum disorder (ASD) to derive biomarkers. Furthermore, P300 decoding can be exploited in BCIs to reinforce ASD impaired skills. Decoders based on convolutional neural networks (CNNs) have been recently proposed, outperforming traditional algorithms but they i) do not investigate the optimal design in different training conditions; ii) lack in interpretability. To overcome these limitations, we propose an interpretable CNN (ICNN) whose optimal design was searched via Bayesian optimization. The ICNN provides a straightforward interpretation of spectral and spatial features learned to decode P300 in ASD. The Bayesian-optimized ICNN design was investigated separately for different training strategies (within-subject, within-session, and cross-subject). Furthermore, transfer learning (TL) potentialities were investigated by assessing how pre-trained cross-subject models performed on a new subject vs. random-initialized models. The ICNN was combined with an Explanation Technique (ICNN+ET) to analyze P300 spectral and spatial features. The ICNN resulted comparable or even outperformed existing CNN approaches, at the same time being lighter and more interpretable. Bayesian-optimized ICNN designs differed depending on the training strategy, needing more capacity as the training set variability increases. Furthermore, TL provided higher performance than networks trained from scratch. The ICNN+ET analysis suggested that the frequency range [2, 5.8] Hz was the most relevant, and spatial features showed a right-hemispheric parietal asymmetry. The ICNN+ET-derived features, but not ERP-derived features, resulted significantly and highly correlated to ADOS clinical scores. This study substantiates the idea that a CNN can be designed both accurate and interpretable for P300 decoding, with an optimized design depending on the training condition. Furthermore, the novel ICNN-based analysis tool was able to better capture and highlight ASD neural signatures compared to a traditional ERP analysis and may pave the way for the identification of novel biomarkers.

3.1. INTRODUCTION

Autism spectrum disorder (ASD) is a set of neurodevelopmental conditions with persistent deficits in social communication and social interaction across multiple contexts, together with restricted, repetitive patterns of behavior, interests, or activities [1]. ASD people show difficulties in social-emotional reciprocity, in developing, maintaining, and understanding relationships, and in non-verbal communicative behaviors used for social interactions, such as joint attention [2–6]. Joint attention emerges during the first year of life and involves the non-verbal coordination of attention of two individuals towards an object or event [7], playing an important role in the development of social and language capabilities [8,9].

Approaches based on neuroimaging, e.g., diffusion tensor imaging and functional magnetic resonance imaging, are used to characterize and identify potential neural biomarkers of information-processing deficits in children with autism [10,11]. In addition, Event-Related Potentials (ERPs) computed from the electroencephalogram (EEG) provide a less expensive and portable way to study sensory information processing and are applied to study the neural response in autism following an incoming stimulus [12]. Investigations concern alterations in both early ERP components reflecting pre-attentive sensory processing and/or initial orientation and capture of attention, such as P100 [13] and N100 [14], and later components, such as P300 [15,16]. The P300 response is characterized by a positive deflection that occurs while attending a target stimulus; it peaks between 250-500 ms after the stimulus onset and it is mostly distributed on the scalp around midline electrode sites (Fz, Cz, Pz), increasing its amplitude from frontal to parietal sites [17]. This response can be elicited in two-stimuli oddball paradigms, where an infrequent (target) stimulus is immersed into a sequence of more frequent (standard) stimulus. The amplitude of the P300 response was found to be positively correlated with allocation of attention resources, stimulus recognition and updating of working memory [18,19]. Abnormalities in the sensory information processing following an incoming stimulus were found in ASD, as quantified by a reduced P300 amplitude in auditory and visuo-spatial tasks compared to healthy subjects, reflecting deficiencies in cognitive, attentional, and working memory processes [12,20–27].

Besides being potentially useful as an EEG-based ASD biomarker, the P300 response can be used for ASD intervention. Indeed, Brain-Computer Interfaces (BCIs) proved to be useful personalized therapeutic approaches in ASD [30–34]; these interfaces can be designed to train ASD people via an EEG-based neurofeedback aimed to reinforce social interactions and communication skills (e.g., joint attention [28,29]). In this scenario, the P300 response elicited in visuo-spatial tasks represents an important control signal for the BCI system [28,29]. A crucial stage of a BCI is represented by the decoding algorithm, that detects the P300 response from the EEG and translates it into a command. Challenges to perform this step arise from the noise sensitivity, non-linearity and non-stationarity of the EEG, as these characteristics depend on the subject and on the environment [33]. In particular, the non-stationarity causes shifts in the EEG across trials and recording sessions. In addition, inter-subject variability across subjects, due to anatomical and physiopathological differences, hinders the design of a ‘participant-agnostic’ BCI. Therefore, due to intra- and inter-subject variabilities, most BCIs require long calibration times on each recording session to achieve satisfactory decoding performance [31]. Furthermore, due to reduced P300 amplitude in ASD, the decoding of the

P300 response is even more challenging and, thus, the decoding performance may be negatively affected.

Machine learning algorithms have been widely adopted to learn discriminative patterns from the EEG to perform P300 decoding [32]. Among these algorithms, traditional decoders for P300-based BCIs perform a pre-processing step that includes band-pass filtering within fixed EEG bands (e.g., 0.5-4 Hz and 4-8 Hz [33]; 2-20 Hz and 2-8 Hz [34]; 2-12 Hz [35]), followed by extraction of features in the temporal, frequency and spatial domains; the latter are then evaluated by a classifier such as linear discriminant analysis, support vector machine or multi-layer perceptron [33,35–40]. In addition to these traditional decoders, significant improvements in performance were found using convolutional neural networks (CNNs) [41–43]. CNNs are feed-forward neural networks including the convolutional operator at least in one layer. Inspired by the hierarchical structure of the ventral stream of the visual system, CNNs are composed by stacked layers of neurons, each neuron characterized by a local receptive field. Neurons in deeper layers have larger receptive field and respond to more complex features [44], enabling the learning of hierarchically structured features from the input signal. At variance with traditional machine learning algorithms, in which a separation between feature extraction, selection, and classification occurs, CNNs solve the decoding task in an end-to-end fashion, by automatically learning the more meaningful features for the addressed problem, i.e., discrimination of P300 events from single EEG trials. Therefore, CNNs do not rely on some characteristics extracted *a priori* from the signals (e.g., spectral contents within fixed bands), but automatically learn the relevant discriminative features to distinguish the P300 response from the input EEG trial. From their first application in P300 decoding with the simple design proposed by Cecotti et al. [45], CNNs were improved by including progressively more convolutional and regularization layers (e.g., dropout [46] and batch normalization [47]) [48]. Among these CNNs, EEGNet [49] and its variants [43,50] were found to be particularly suitable for P300 decoding, outperforming traditional machine learning solutions as well as other CNN-based approaches, also in case of P300-based BCIs aimed at ASD intervention [41,43]. In addition, by adopting specific training strategies (such as within-subject and cross-session strategy or cross-subject strategy), CNNs were found to be capable of learning robust features across sessions and subjects, incapsulating intra-subject and inter-subject variability, thus providing the potentiality of significantly reducing BCI calibration times [43].

However, despite the previous advantages, these algorithms have some limitations. First, they introduce a large number of trainable parameters, i.e., parameters to fit during the training process, and require setting many hyper-parameters, i.e., parameters that define the functional form of the decoder (e.g., convolutional filter size, number of convolutional filters, type of activation function, etc.) that must be set before the training.

Generally, hyper-parameters are selected by testing only a few configurations via empirical evaluations [43,48–50], and, thus, CNNs are built with a sub-optimal design in terms of performance. Furthermore, hyper-parameters are kept the same across different training strategies [43,50]. In this way, the network capacity (i.e., its ability of approximating a wide variety of functions) is kept the same even though the network faces problems with increasing difficulty across the different strategies, e.g., when moving from within-subject and within-session to cross-session and cross-subject decoders, the architecture must learn the relevant features from training distributions with increased variability. Searching for an optimal CNN

hyper-parameter configuration, which is also specific for a particular training strategy, is even more necessary when designing a P300 decoder for a BCI-based ASD therapeutic approach, where the P300 response is attenuated and more difficult to distinguish than in case of healthy users.

Moreover, CNNs are scarcely interpretable in their learned features and are often treated as black boxes. As pointed out in a recent survey [51], there is a growing interest to interpret the feature representations provided by deep neural networks and their relation to clinical outcomes quantifying neuropathology. Thus, research is moving from the sole decoding of brain states towards the analysis of EEG features derived from the learning system, for example studying EEG signatures related to movement [52–56], P300 [43,57,58], or depression [59]. Recently, interpretable CNNs (ICNNs), i.e., CNNs that include layers whose learned parameters are directly interpretable, were proposed to decode motor imagery and execution. Zhao et al. [53] increased feature interpretability in the frequency domain by reparametrizing a convolutional layer to learn Morlet wavelets. Recently, we proposed a lightweight (i.e., with a limited number of trainable parameters) ICNN [52,55] able to increase the interpretability of both spectral and spatial features, by reparametrizing a convolutional layer to learn band-pass filters and by learning spatial filters tied to each band-pass filter. In addition to interpretable layers, explanation techniques (ET) can be used to explain the CNN decision by highlighting which EEG features learned in interpretable domains (e.g., spatial, temporal, spectral) are the most discriminative (i.e., most salient) for the decoding of a specific cognitive state, such as the P300 [43,57,58]. Thus, by leveraging on the increased interpretability embedded into an ICNN, combined with an ET (denoted as ICNN+ET in the following), a data-driven non-linear analysis tool could be realized able to gain insights into the physiopathological neural signatures contained in the EEG associated to P300.

In this study, we aim to contribute to the decoding and, at the same time, to the analysis of the P300 in ASD using an ICNN, to further increase the feasibility of BCI intervention in autism and to potentially characterize novel data-driven biomarkers related to ASD visuo-spatial sensory processing. To this aim, here we adopted an architecture obtained by combining two existing CNN architectures: Sinc-ShallowNet [52], i.e., our previous interpretable and lightweight design (but previously proposed for decoding tasks other than P300), which allows a straightforward interpretation of spectral and spatial features learned by the ICNN, and EEGNet, which represents the state-of-the-art (SOA) for P300 decoding [41,50]. Then, using this architecture, we address the following two issues as main points of novelty of this study:

i. Investigate the optimal ICNN design (in terms of performance) and the role of the main ICNN hyper-parameters under different training conditions, by performing automatic hyper-parameter search (AHPS) based on Bayesian optimization (BO), separately for each training condition. The investigated training strategies were within-subject and within-session, within-subject and cross-session, and leave-one-subject-out. Furthermore, we also tested the capability to transfer the knowledge from other participants to a new one, adopting a transfer learning strategy to reduce calibration time. Each of these training strategies may represent a different practical scenario of BCI intervention to improve joint attention in ASD.
ii. Design of a novel algorithm based on the combination of the ICNN with saliency representations (ICNN+ET) to highlight the most relevant spectral and spatial features

that correspond to the visuo-spatial P300 correlate in autism. These features were then used to define clusters of subjects characterized by shared neural signatures. Then, we investigated whether these features were related to clinical scores measuring the severity of ASD symptoms as derived by developmental and behavioral ASD assessment tools, and whether the ICNN+ET analysis better enhanced useful ASD neural signatures compared to a canonical ERP analysis.

3.2. MATERIALS AND METHODS

3.2.1. Dataset description

In this study we used the BCIAUT-P300 dataset, an EEG dataset of a feasibility clinical trial [28,29] publicly released for the IFMBE 2019 scientific challenge (https://www.kaggle.com/disbeat/bciaut-p300). EEG signals were recorded from 15 high-functioning ASD participants (22 years old, on average) while testing a P300-based BCI (based on statistical classifiers) aimed to improve their joint attention. During a baseline visit, the following clinical scores, useful also to diagnose ASD, were collected: Autism Diagnostic Observation Schedule (ADOS) [60], Autism Diagnostic Interview-Revised (ADI-R) [61], and Intelligence Quotients (IQs, measured by WAIS-III [62]). These scores are reported in Table 3.1.

Table 3.1 – Measured ASD clinical scores, i.e., ADOS, ADI-R, and IQs (mean ± standard deviation across subjects).

Measure	Value
ADOS A: Communication	3.20±0.86
ADOS B: Social Interaction	6.27±1.29
ADOS A+B: Communication-Social Interaction	9.47±1.86
ADI-R 1: Social Interaction	16.14±4.39
ADI-R 2: Communication and Language	12.14±5.19
ADI-R 3: Restricted and Repetitive Behavior	6.14±2.33
FSIQ: Full Scale IQ	102.53±11.24
VIQ: Verbal IQ	102.33±16.06
PIQ: Performance IQ	102.46±10.59

For each participant, the BCI paradigm was carried out during 7 recording sessions (over 4 months) and in each session it was based on an immersive virtual environment presented to the user, consisting of a bedroom with an avatar, common furniture (e.g., shelves, a bed, etc.) and eight objects of interest. These objects were: 1) books on a shelf, 2) a radio on top of a dresser, 3) a printer on a shelf, 4) a laptop on a table, 5) a ball on the ground, 6) a corkboard on the wall, 7) a wooden plane hanging from the ceiling, and 8) a picture on the wall (see Figure 3.1).

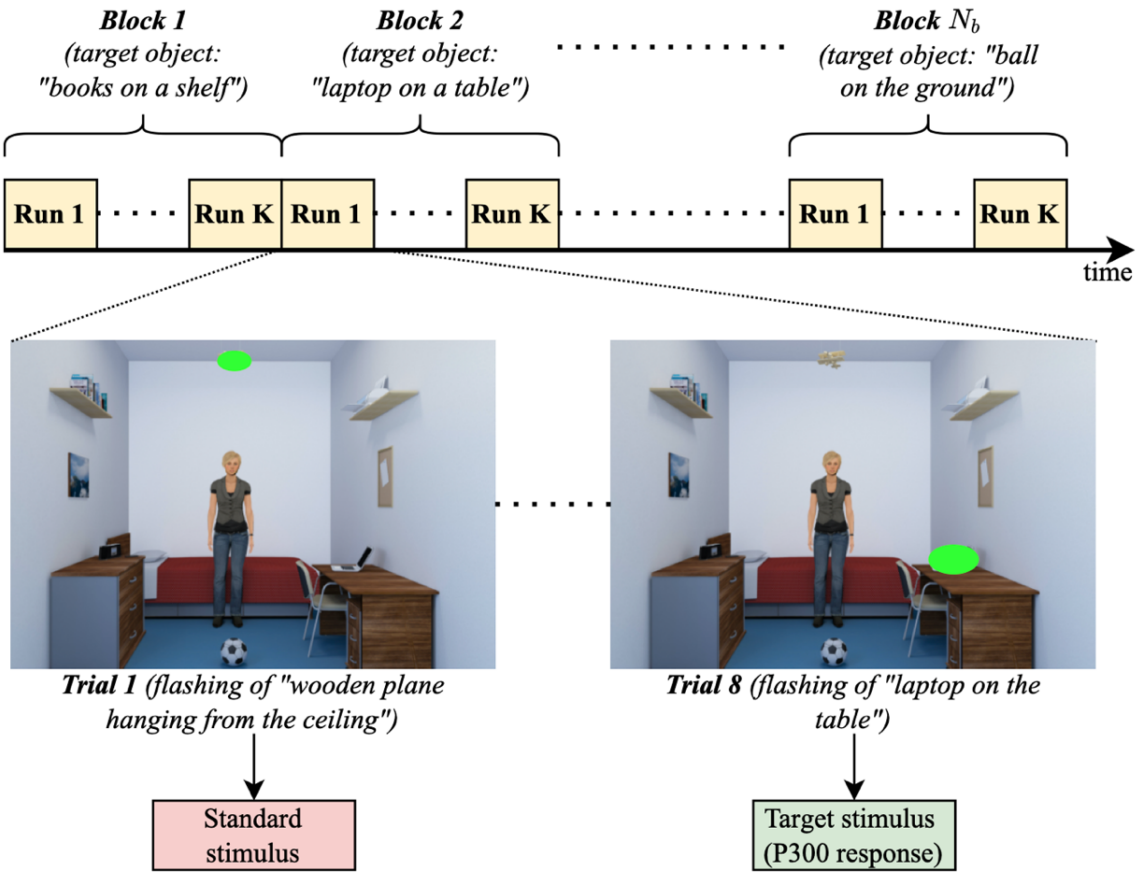


Figure 3.1 – Structure of the BCI paradigm in its division into blocks, runs, and trials. Each recording block was used to identify one of the eight target objects visualized in the virtual environment, with a total of $N_b = 20$ (calibration phase) and of $N_b = 50$ (online phase) blocks. Within each block, K runs were recorded, where $K = 10$ (calibration phase) and $K = 7.095$ (online phase) on average across subjects and sessions. In each run, the 8 objects randomly flashed (schematized by the green ellipses in the figure) and 8 EEG trials per run where recorded. Therefore, overall, within each recording session $N_b \cdot K \cdot 8$ EEG trials were recorded, corresponding 1600 trials in the calibration phase and 2838 trials on average in the online phase, respectively.

Participants were instructed to read non-verbal social agent cues from the avatar, i.e., avatar head turning to a particular object, and to pay attention to that target object. The 8 objects randomly flashed in the virtual scene and, thus, a visuo-spatial P300 response was elicited when the attended object flashed.

Each recording session was composed by a calibration phase, consisting of $N_b = 20$ blocks, and an online phase, consisting of $N_b = 50$ blocks. Each block was related to a particular object selected as target by the avatar that the user tried to identify (see Figure 3.1 for a scheme about the organization of blocks, runs, and trials). For each block, K runs were repeated ($K = 10$ in the calibration phase, while K varied in the online phase with $K = 7.095$ on average across online blocks); each run consisted of each of the 8 objects flashing once in a random sequence. This resulted in 1600 trials (20 blocks x 10 runs x 8 EEG trials) per each participant and session during the calibration phase, and in 2838 trials on average during the online phase. During the calibration phase, the BCI statistical classifiers were trained to predict the object the participant was paying attention to; during the online phase, the trained classifiers were applied to identify

whether the subject attended the target object correctly and, in that case, to provide positive feedback to the user.

The IFMBE 2019 scientific challenge was organized to stimulate researchers to develop decoders maximizing the object detection accuracy based on the P300 response. In the challenge, for each subject and each session, trials recorded during the calibration phase were released to *tune* decoders (i.e., to set their parameters), while trials recorded during the online phase were used to *test* decoders. In the present study, we adopted this same split as defined by the challenge, as this choice may also allow a fair and direct comparison with the results of decoders that participated to the challenge [41]. More specifically, for each subject and each session, we further split the 1600 calibration trials into two sets: 80% of trials (1280 trials), were randomly sampled and used as *training set* to optimize the trainable parameters of the decoders, while the remaining 20% of trials (320 trials) were used as *validation set* to optimize the hyper-parameters of the decoders. This further splitting was based on our winning solution of the challenge [41,50] to select the number of training epochs (i.e., to perform early stopping). Therefore, each subject-specific and session-specific dataset was separated into 3 different sets, as commonly performed in the literature [41,43,45,49,50,52,55,57,63]: training set (1280 trials), validation set (320 trials), and test set (2838 trials on average). However, since different training strategies were adopted here, by differently aggregating the training set and validation set across sessions and subjects, the overall number of training and validation trials were different depending on the strategy (see Section *Training strategies*). Finally, test trials were used to test the tuned decoders on a held-out set.

EEG signals were recorded at 250 Hz from C3, Cz, C4, CPz, P3, Pz, P4, and POz locations ($C = 8$ electrode sites), with the reference placed at the right ear and the ground at AFz. These signals were acquired notch filtered at 50 Hz and filtered between 2 and 30 Hz [61]. In this study, as in the winning solution [50] that we proposed for the challenge, each trial contained signals from -0.1 s to 1 s relative to the stimulus onset and signals were downsampled to 128 Hz, so that each trial contained $T = 140$ time steps.

3.2.2. Problem definition

Based on the previous description, the collection of trials associated to the s-th subject acquired during the r-th recording session can be formalized as the collection $D^{(s,r)} = \left\{\left(X_0^{(s,r)}, y_0^{(s,r)}\right), \dots, \left(X_i^{(s,r)}, y_i^{(s,r)}\right), \dots, \left(X_{M^{(s,r)}-1}^{(s,r)}, y_{M^{(s,r)}-1}^{(s,r)}\right)\right\}$. $M^{(s,r)}$ denotes the total number of trials for that subject and that session, $X_i^{(s,r)} \in \mathbb{R}^{C \times T}$ represents the pre-processed EEG signals of the i-th trial ($0 \leq i \leq M^{(s,r)} - 1$), and $y_i^{(s,r)}$ represents the label of the i-th trial, i.e., $y_i^{(s,r)} \in L = \{l_0, l_1\} = \{\text{non-P300, P300}\}$, where the label P300 was assigned to those trials where the flashing object coincided with the object the avatar was looking at.

In this study, each decoder consisted in a parametrized classifier f (representing the ICNN and having a different functional form depending on the hyper-parameters), which solves a binary classification task: $f\left(X_i^{(s,r)}; \theta\right): \mathbb{R}^{C \times T} \rightarrow L$, where θ represents the array of trainable parameters. Thus, the ICNN input was represented by $X_i^{(s,r)}$ that can be viewed as a 2D matrix of shape $(C, T) = (8,140)$ with electrodes along the height and time steps along the width, and

the output consisted of two neurons corresponding to either one or the other class. The validation set was used to perform the automatic search of ICNN hyper-parameters via Bayesian optimization, and the ICNN learned automatically from the training set the relevant features to assign the correct label to unseen input trials belonging to the test set. The knowledge learned by the ICNN during the training process was stored in its trainable parameters θ. These parameters, thanks to their increase interpretability, can be exploited to gain insights into the neural signatures related to a specific class (e.g., P300 class) in a data-driven way, without relying on handcrafted features based on expected EEG responses.

3.2.3. An update of Sinc-ShallowNet: Sinc-ShallowNet-v2

Sinc-ShallowNet [52] is an ICNN that we developed to decode motor imagery and execution from single EEG trials. This ICNN is composed of two blocks, each consisting of several stacked layers: an interpretable spectral and spatial (ISS) feature extractor followed by a fully-connected (FC) block that performs classification. The ISS block was designed to increase the interpretability of the learned parameters, at the same time keeping limited the model size, i.e., the number of trainable parameters, and included a temporal convolutional layer (with a reparameterization of the kernels), a depthwise spatial convolution and an averaging pooling layer (see also below for a more detailed description of the ISS block). Crucially, in Sinc-ShallowNet the output of the ISS block was provided directly as input to the FC block. Here, we proposed an updated version for P300 decoding, named Sinc-ShallowNet-v2, by integrating in the design also structure elements of EEGNet, a CNN proved to be particularly suitable to decode the P300 response from EEG [41,49,50] but not designed to be interpretable. Specifically, the updated Sinc-ShallowNet-v2 embraces three main blocks by including an additional block, the fixed-scale temporal (FST) feature extractor (inspired by EEGNet), between the ISS block and the FC classification block. This is an important modification, since Sinc-ShallowNet-v2 further processes the output of the ISS block by learning features in the temporal domain, and this may help to better capture intra- and inter-subject variability of P300 in time. A schematization of Sinc-ShallowNet-v2 is reported in Figure 3.2a; a more detailed description reporting the hyper-parameters, output shape and number of trainable parameters per layer, is reported in Table 3.2. In the following, the blocks are described in detail.

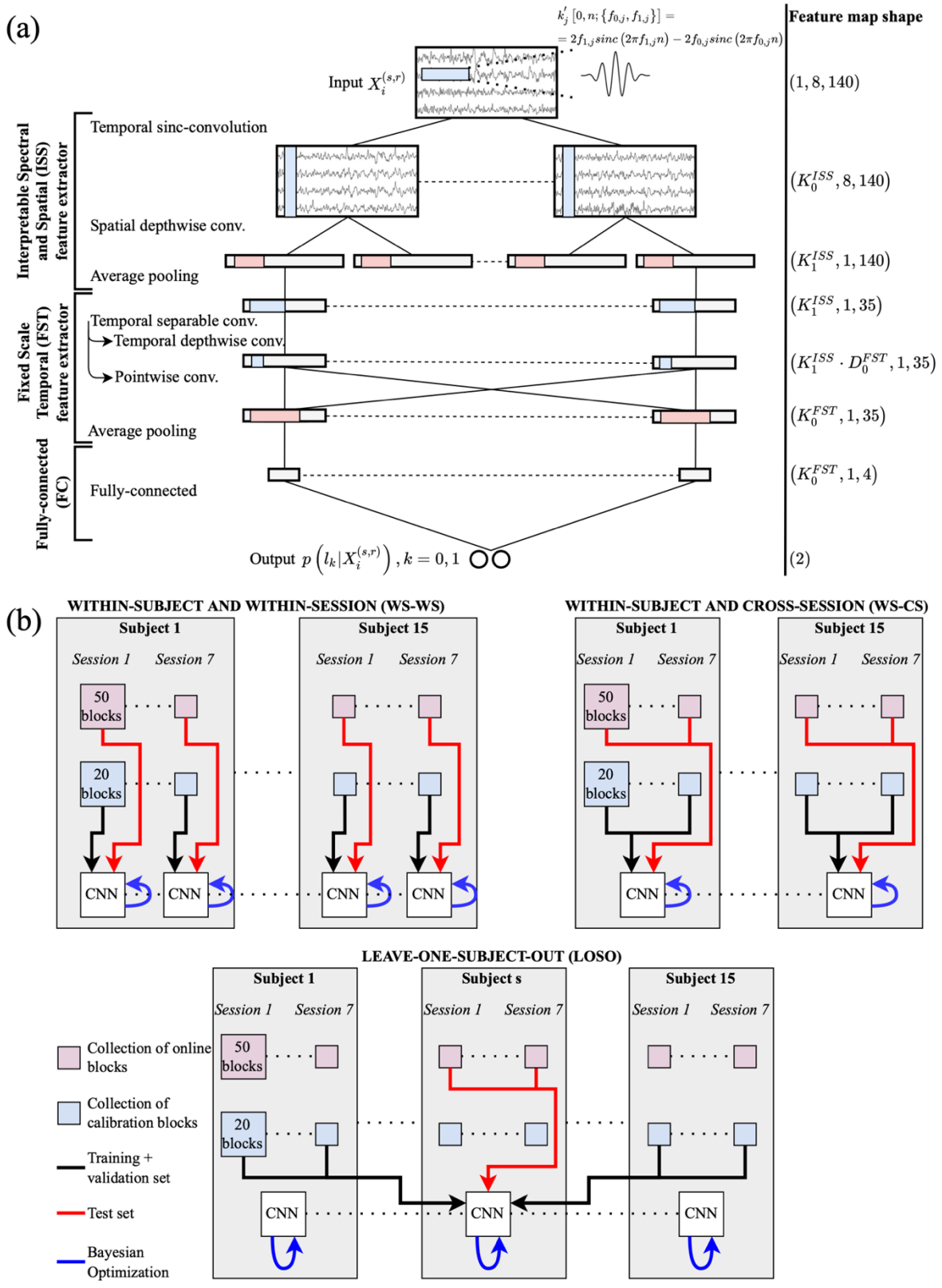


Figure 3.2 – Sinc-ShallowNet-v2 (a) and training strategies investigated while performing Bayesian optimization (b). In Figure 3.2a, blocks and main layers are listed on the left side. To keep the figure as clear and simple as possible, only the main trainable layers (i.e., convolutional and fully-connected layers) in addition to pooling layers (to highlight the temporal dimension reduction) were displayed. Boxes represent the output feature maps of each layer, and colored rectangles represent convolutional (blue) and pooling (red) kernels. Tuples reported on the right side represent the shape of the feature maps. For all outputs except the last, tuples are composed by three numbers representing the number of the feature maps, the number of spatial samples and the number of temporal

samples within each map, respectively. The input layer provides an output of shape $(1, C, T) = (1,8,140)$, as it just replicates the original input EEG trial with shape $(8,140)$, providing a single feature map as output. The temporal dimension changed from $T = 140$ to $T//4 = 35$ and to $T//32 = 4$ along the entire CNN due to the average pooling operations (where // indicates the floor division operator). See Section 3.2.2, 3.2.3 and Table 3.2 for further details. Figure 3.2b shows a schematic representation of how training, validation, and test examples (by means of black and red arrows) were sampled from calibration (blue boxes) and online (purple boxes) blocks recorded in the BCI paradigm when training decoders in Bayesian optimization in within-subject and within-session (WS-WS), within-subject and cross-session (WS-CS), and leave-one-subject-out (LOSO) training conditions. For brevity, in the LOSO strategy the aggregation across blocks is reported only for the s-th subject. See Section 3.2.1 and Section *Training strategies* for further details about the definition of recording blocks and the training strategies, respectively.

Table 3.2 – Sinc-ShallowNet-v2. Each layer is provided with its name, main hyper-parameters and number of trainable parameters. See Sections 3.2.2 and 3.2.3 for the meaning of the symbols. In all layers, where not specified, stride (S) and padding (P) were set to $(1,1)$ and $(0,0)$, respectively. The structural hyper-parameters (i.e., hyper-parameters of the architecture structure) reported in bold were searched via Bayesian optimization.

Block	Layer name	Hyper-parameters	Number of trainable parameters
	Input	$K_0 = 1$	0
ISS	Sinc-Conv2D	$\boldsymbol{K_0^{ISS}}$, ${F_0}^{\text{ISS}} = (1,65)$, ${P_0}^{\text{ISS}} = (0, {F_0}^{ISS}[1]//2)$	$2 \cdot {K_0}^{ISS} \cdot K_0$
	BatchNorm2D		$2 \cdot {K_0}^{ISS}$
	Depthwise-Conv2D	$\boldsymbol{D_1^{ISS}}$, ${K_1}^{ISS} = {K_0}^{ISS} \cdot {D_1}^{ISS}$, ${F_1}^{ISS} = (C, 1)$, $\boldsymbol{c_1^{ISS}}$	${F_1}^{ISS}[0] \cdot {F_1}^{ISS}[1] \cdot {K_1}^{ISS}$
	BatchNorm2D		$2 \cdot {K_1}^{ISS}$
	ELU		0
	AvgPool2D	${F_p}^{ISS} = {S_p}^{ISS} = (1,4)$	0
	Dropout	$\boldsymbol{p^{ISS} = p^{FST} = p}$	0
FST	Separable-Conv2D	$\boldsymbol{K_0^{FST}}$, $\boldsymbol{F_0^{FST}}$, ${D_0}^{FST} = 1, {P_0}^{FST} = (0, {F_0}^{FST}[1]//2)$	${F_0}^{FST}[0] \cdot {F_0}^{FST}[1] \cdot {K_1}^{ISS} + {K_1}^{ISS} \cdot {K_0}^{FST}$
	BatchNorm2D		$2 \cdot {K_0}^{FST}$
	ELU		0
	AvgPool2D	${F_p}^{FST} = {S_p}^{FST} = (1,8)$	0
	Dropout	$\boldsymbol{p^{FST} = p^{ISS} = p}$	0
FC	Flatten		0
	Fully-Connected	$N^{FC} = 2$, $\boldsymbol{c_0^{FC}}$	$N^{FC} \cdot (T//32 \cdot {K_0}^{FST} + 1)$
	Softmax		0

i. Block 1: Interpretable spectral and spatial (ISS) feature extractor

The first block was inspired by the ISS block of Sinc-ShallowNet [52] and was devoted to separately learn spectral and spatial features from the input EEG trials in an easy interpretable way. The very first layer of the ISS block was a temporal sinc-convolutional layer [52,55,64], learning ${K_0}^{ISS}$ filters with filter size ${F_0}^{ISS} = (1,65)$, unitary stride and zero-padding ${P_0}^{ISS} = (0,32)$ to preserve the number of input temporal samples. This temporal convolutional layer was devoted to filter each electrode signal in time. By using the sinc-convolutional layer to perform such processing step instead of a conventional convolutional layer, each convolutional filter is forced to describe a band-pass filter in the temporal domain. Denoting with k_j the j-th

convolutional kernel, in a conventional convolutional layer each value of the filter (i.e., $k_j[0,n], n \in [0,64]$) has to be learned during the optimization process; conversely, in a sinc-convolutional layer, each value of the filter is defined by a parametrized function, forcing the filters to belong to a specific subset of temporal filters (here band-pass filters). Therefore, in a sinc-convolutional layer a re-parametrization of each convolutional kernel occurs:

$$k_j'[0,n;\{f_{0,j},f_{1,j}\}] = 2f_{1,j}sinc(2\pi f_{1,j}n) - 2f_{0,j}sinc(2\pi f_{0,j}n). \quad (3.1)$$

In Equation 3.1, $\{f_{0,j},f_{1,j}\}$ are the trainable parameters related to the j-th kernel, including only the inferior ($f_{0,j}$) and superior ($f_{1,j}$) cutoff frequencies of the band-pass filter. In this way, the number of trainable parameters reduces from 65 ($= {F_0}^{ISS}[0] \cdot {F_0}^{ISS}[1]$) to 2, for each temporal filter. Lastly, to alleviate the effects of the inevitable truncation of k_j' on the characteristics of each filter, k_j' is multiplied by a Humming window:

$$\begin{cases} {k_{w,j}}'[0,n;\{f_{0,j},f_{1,j}\}] = k_j'[0,n;\{f_{0,j},f_{1,j}\}] \cdot w[n] \\ w[n] = 0.54 - 0.46\cos\left(\frac{2\pi n}{{F_0}^{ISS}[1]-1}\right) \end{cases}. \quad (3.2)$$

Accordingly, the temporal sinc-convolution computes the convolution between the input and ${k_{w,j}}'[0,n;\{f_{0,j},f_{1,j}\}]$, learning only the following 2 parameters for each kernel:

$$\theta_{spect,j} = \{f_{0,j},f_{1,j}\} \in \theta, 0 \le j \le {K_0}^{ISS} - 1. \quad (3.3)$$

Thus, the output of this first layer consists of stacked feature maps containing band-pass filtered versions of the input EEG trial within specific frequency ranges that were explicitly learned during training.

The use of a temporal sinc-convolutional layer, besides substantially reducing the parameters to fit, promotes the learning of more meaningful and well-defined temporal filters, and provides a straightforward interpretation of the learned spectral features [64], being the cutoff frequencies of the learned band-pass filters. Conversely, in case of a conventional temporal convolutional layer, the learned spectral features are not immediately accessible and interpretable.

Downstream the temporal sinc-convolutional layer, a batch normalization layer [47] was included. Then, a spatial depthwise convolutional layer was introduced: for each band-pass filtered map, ${D_1}^{ISS}$ spatial filters were learned having size $(C, 1)$, unitary stride and no zero-padding, i.e., ${D_1}^{ISS}$ spatial combinations of electrodes were learned for each band-pass filtered map. Therefore, a total number of ${K_1}^{ISS} = {K_0}^{ISS} \cdot {D_1}^{ISS}$ spatial filters were learned and constrained to have a norm upper bounded by ${c_1}^{ISS}$ (kernel max-norm constraint). This type of convolution does not exploit dense connections across feature maps as in traditional convolutional layers, reducing the number of trainable parameters. In addition, the combination of temporal sinc-convolution with spatial depthwise convolution leads to an interpretable spectral-spatial feature learning, as each group of ${D_1}^{ISS}$ spatial filters is strictly tied (via depthwise convolution) to a specific band-pass filter, i.e., to a specific frequency range:

$$\theta_{spat,j} = \{\theta_{j0}, \dots \theta_{jk}, \dots, \theta_{j{D_1}^{ISS}-1}\} \in \theta, 0 \le j \le {K_0}^{ISS} - 1, \quad (3.4)$$

indicating with θ_{jk} the k-th spatial filter ($0 \le k \le {D_1}^{ISS} - 1$) tied to the j-th band-pass filter.

The parameters learned in these two first convolutional layers can be denoted as the set:

$$\theta_{ISS} = \left\{\theta_{ISS,0}, \dots, \theta_{ISS,j}, \dots, \theta_{ISS,K_0{}^{ISS}-1}\right\} = \left\{\left(\theta_{spect,0}, \theta_{spat,0}\right), \dots, \left(\theta_{spect,j}, \theta_{spat,j}\right), \dots, \left(\theta_{spect,K_0{}^{ISS}-1}, \theta_{spat,K_0{}^{ISS}-1}\right)\right\}, \quad (3.5)$$

where each pair $\left(\theta_{spect,j}, \theta_{spat,j}\right)$, $0 \leq j \leq K_0{}^{ISS} - 1$ contains the cutoff frequencies of the j-th band-pass filter and the associated $D_1{}^{ISS}$ spatial filters exploited to decode the input EEG trial.

Output activations of the interpretable spectral-spatial feature extractor were then normalized via batch normalization [47] and activated via an Exponential Linear Unit (ELU) non-linearity [65], i.e., $f(x) = x, x > 0$ and $f(x) = exp(x) - 1, x \leq 0$. Average pooling was introduced to reduce the temporal dimension of the activations by a factor of 4 ($F_p{}^{ISS} = S_p{}^{ISS} = (1,4)$) from $T = 140$ to $T//4 = 35$ (indicating with // the floor division operator). Lastly, a dropout layer [46] was added with dropout rate p^{ISS}.

ii. Block 2: Fixed-scale temporal (FST) feature extractor

This block was inspired by EEGNet, and in Sinc-ShallowNet-v2 was used to learn how to summarize the feature maps provided by the ISS block in the time domain; it contained a separable convolutional layer [66], defined by a temporal depthwise convolution with $D_0{}^{FST} = 1$, filter size $F_0{}^{FST}$, unitary stride and zero padding $P_0{}^{FST} = (0, F_0{}^{FST}[1]//2)$, followed by a pointwise convolution with $K_0{}^{FST}$ filters. In the first layer of this composition, depending on $F_0{}^{FST}$, temporal features were learned on the input ISS feature maps within a temporal window of specific size, i.e., at a fixed temporal scale, and without using dense connections across feature maps. The temporal window in which features are learned corresponds to $F_0{}^{FST}[1]/\,(sf//4)$ s, indicating with sf the sampling frequency of the input EEG signals. In the second layer of the separable convolutional, feature maps at the output of the previous layer are recombined, and, this is performed learning compressed, equal or overcomplete representations, depending on whether $K_0{}^{FST} < K_1{}^{ISS}$, $K_0{}^{FST} = K_1{}^{ISS}$ or $K_0{}^{FST} > K_1{}^{ISS}$, respectively. Then, activations were normalized via batch normalization [47] and activated via an ELU non-linearity [65]. An average pooling layer was introduced to further reduce the temporal dimension of the activations by a factor of 8 ($F_p{}^{FST} = S_p{}^{FST} = (1,8)$), i.e., from $T//4 = 35$ to $T//32 = 4$. Lastly, a dropout layer [46] was added with dropout rate p^{FST}.

iii. Block 3: Fully-connected (FC) block

This block traduces the activations provided by the FST block into conditional probabilities, finalizing the decoding task. At first, the input feature maps of the FC block were unrolled along one single dimension using a flatten layer, producing a single feature array. Then, this array was provided as input to a single fully-connected layer with $N^{FC} = 2$ neurons associated to the non-P300 and P300 classes. Here, trainable parameters were constrained to have a norm upper bounded by $c_0{}^{FC}$. Lastly, these 2 neurons were activated using a softmax activation function to obtain the output conditional probabilities $p\left(l_k \middle| X_i^{(s,r)}\right)$, $k = 0, 1$.

3.2.4. Performance metric and comparison of Sinc-ShallowNet-v2 with EEGNet and Sinc-ShallowNet

In this study, we adopted the object-level accuracy as performance metric, i.e., accuracy in decoding the flashing object the participant was paying attention to (among the 8 possible objects, see Section 3.2.1 for additional details). This performance metric was the same as adopted in the IFBME 2019 competition to evaluate decoders [41]. To this aim, the trial-level EEG decoding provided by the CNN (binary classification, i.e., non-P300 vs. P300) was used to produce an object-level decoding (8-class classification, i.e., discrimination of the attended object among 8 objects). Considering a specific recording block, associated to a specific object the participant was paying attention to, the following processing was performed. Indicating with l_1 the P300 condition, we considered the probabilities $p\left(l_1 \middle| X_i^{(s,r)}\right)$ predicted for the EEG trials $X_i^{(s,r)}$ within that block when each of the 8 objects flashed (including the attended one). These probabilities where averaged across runs separately for each object, obtaining the average probability $\bar{p}_o, 0 \leq o \leq 7$ that the participant was paying attention to the o-th object Then, the object with the highest probability was predicted as the one attended in that block.

We compared the performance of Sinc-ShallowNet-v2 against our approach derived from EEGNet [50] that won the IFMBE 2019 challenge. In the competition, that approach significantly outperformed traditional machine learning solutions as well as other deep neural networks, including both CNNs [48] and recurrent neural networks [41], thus, it can be considered the current SOA algorithm for the addressed decoding problem. Furthermore, Sinc-ShallowNet-v2 performance was compared against Sinc-ShallowNet [52], to assess whether the inclusion of the FST block in the updated version led to potential benefit in P300 decoding compared to the previous architecture (lacking this block).

3.2.5. Trainable parameter optimization

Training strategies

The structure of the adopted dataset BCIAUT-P300 (including several sessions per subject and a large number of trials per session), allows to train and evaluate P300 decoders in a large spectrum of training conditions, each reflecting a different scenario in which a P300-based BCI intervention might be applied. In this study, we assessed the optimal design of the proposed ICNN for P300 decoding, separately for each of different training strategies, characterized by increasing variability in the dataset, i.e., simulating BCI paradigms applied progressively to more sessions and subjects.

As introduced in Section 3.2.1, the trials of each session and subject were divided into training validation sets (80% and 20% of trials of the calibration phase, randomly sampled) and test set (trials of the online phase). Then, depending on whether training and validation sets were considered separately for each subject or session, or were differently aggregated across sessions and subjects, four different training strategies were designed and investigated. It is worth mentioning that, despite different training strategies were employed, the definition of the test set was the same across training strategies to perform a fair comparison across them, that is, the test set always included trials belonging to the 50 online blocks. These strategies are described in the following. Furthermore, Figure 3.2b provides a schematization of the adopted

strategies, and Table 3.3 a summary of the number of trials belonging to the training, validation and test sets in each strategy.

Table 3.3 –Part A: The number of trials in the training set and validation set used to tune the models (under Bayesian optimization) and number of trials in the test set used to test the models' performance, for the different training strategies. Part B: The number of trials in the training set, validation set and test sed used to tune and to test the WS-WS models and the TL-WS models, to evaluate the beneficial effect of transfer learning. Note that in the computational experiments of part B, the validation set was used only for early stopping, as the other hyper-parameters were inherited from the LOSO models tuned in experiments of part A. The trials in the test set were 2838 (on average) in each training strategy, as each model was tested separately on each session-specific test set, and then the performance was averaged across all sessions for each specific subject.

	Training strategy	No. of trials in the training set	No. of trials in the validation set	No. of trials in the test set
A	*Within-subject and within-session (WS-WS)*	1280	320	2838
	Within-subject and cross-session (WS-CS)	8960	2240	2838
	Leave-one-subject-out (LOSO)	125440	31360	2838
B	*Within-subject and within-session (WS-WS)*	From 128 to 1280 (step of 128)	From 32 to 320 (step of 32)	2838
	Transfer learning on single session (TL-WS)	From 128 to 1280 (step of 128)	From 32 to 320 (step of 32)	2838

i. Within-subject and within-session (WS-WS) strategy. In this strategy, subject- and session-specific training and validation sets were used to tune subject- and session-specific CNNs. The test set was defined as the subject- and session-specific test set, i.e., trials of the online phase belonging to the subject and session the CNN was trained for (see Table 3.3A). Furthermore, to simulate a practical scenario where limited trials are available in a single recording session, we trained and validated CNNs using a progressively increasing number of calibration blocks (see Section 3.2.1), and thus, using variable-sized training and validation sets. This condition was implemented by sampling training and validation examples from the first n calibration blocks, where n was progressively increased from 2 to 20 with a step of 2 blocks (see Table 3.3B), where 20 is the overall number of calibration blocks in each session.

ii. Within-subject and cross-session (WS-CS) strategy. In this strategy, subject-specific training and validation sets were used to tune subject-specific CNNs. For the s-th subject, training and validation sets were merged across all recording sessions of that specific subject (see Table 3.3A). incorporating within-subject variability. The architecture was then tested separately on each subject- and session-specific test set, i.e., on trials of the online phase of each session belonging to the subject the CNN was trained for.

iii. Leave-one-subject-out (LOSO) strategy. In this strategy, cross-subject training and validation sets were used to tune cross-subject, cross-session and subject-agnostic CNNs, i.e., the training and validation sets included only examples sampled from other subject distributions. In particular, for each subject s, named "held-back subject", training and

validation sets were merged across all sessions from all the other subjects (different from the s-th subject, see Table 3.3A), incorporating between-subject and within-subject variability. The architecture was then tested separately on each subject- and session-specific test set, i.e., on trials of the online phase of each session belonging to the held-back subject (s-th subject).

iv. Transfer learning on single sessions (TL-WS). Transfer learning focuses on transferring the knowledge across domains/tasks. It is inspired by the human capability to use the knowledge learned in a source domain/task to improve the performance and/or reduce the training time in a related domain/task [67]. In this strategy, as for the WS-WS strategy (Section *Training strategies*-i), training and validation trials were subject- and session-specific. In particular, the definition of training and validation sets was the same as the one adopted in the WS-WS strategy when analyzing variable-sized training and validation sets (depending on the number of included calibration blocks), while keeping unchanged the subject- and session-specific test set, i.e., the trials of the online phase of that subject and session (see Table 3.3B). However, differently from the WS-WS strategy where the trainable parameters were initialized randomly, in the TL-WS strategy, for the s-th subject and r-th recording session the CNN was initialized using the CNN trained with LOSO strategy when the s-th subject was held-back. That is, the knowledge learned during the LOSO strategy from many subjects except the held-back one was transferred on the held back subject. The use of TL-WS strategy could be useful when a new user approaches the BCI system in a new recording session and a calibration stage, as short as possible, is needed to tune an accurate decoder. In this view, the knowledge embedded in LOSO models, i.e., incorporating between-subject and within-subject variabilities, could represent a better initialization point in the space of parameter θ than the random one, potentially leading to an improvement in performance and/or to a reduction of training and validation examples needed to achieve high performance. The potentialities of transfer learning were tested by comparing the performance of the TL-WS models with the performance of the WS-WS models, while increasing the size of the training and validation sets, thus, assessing the benefits of the use of models pre-trained on other subjects compared to models trained from scratch.

Training settings

CNN optimization consisted of the minimization of the cross-entropy between the empirical probability distribution defined by the training labels, and the probability distribution defined by the model. This corresponds to minimize the Kullback-Leibler divergence between the two probability distributions at trial-level, and thus also at block-level, i.e., object-level. Adaptive moment estimation (Adam) [68] was used as optimizer with learning rate lr, mini-batch size $bs = 64$, $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for computing the running averages of the gradient and its square, and $\varepsilon = 10^{-8}$ to improve numerical stability. To address class imbalance, parameter updates were weighted depending on the class occurrence of the training examples: indicating with M_0 and M_1, the number of trials for non-P300 and P300 conditions in the training set and given that $M_0 > M_1$, class weights were defined as 1 and $M_0/M_1 = 7$, respectively for non-P300 and P300 classes. The maximum number of epochs was set to 1000 and early stopping

was performed by interrupting the training loop when the validation loss did not decrease for 50 consecutive epochs. Besides early stopping, which acts as regularizer, Sinc-ShallowNet-v2 implicitly included in its structure also many layers devoted to increase the generalization, such as batch normalization [47] (with a momentum term $m = 0.99$ and $\varepsilon =$ 1e-3 for numerical stability), dropout [46] and kernel max norm constraint.

3.2.6. Hyper-parameter optimization

Hyper-parameter optimization is devoted to find the optimal hyper-parameter configuration of a learning system associated with the best performance measured on a separate validation set. In the following, an overview of hyper-parameter tuning via Bayesian optimization (BO) [69] is provided; then, the procedure adopted to perform BO hyper-parameter search and to report the obtained results is described.

Hyper-parameter search via Bayesian optimization

Denoting with h the array containing the hyper-parameters of interest ($h \in H$, where H is the hyper-parameter search space), the aim of hyper-parameter optimization is to find $h^* = arg \min_{h \in H} k(h)$, that minimizes the objective function $k(h)$ (the cross-entropy loss, in this study) evaluated on the validation set. However, the evaluation of the objective function $k(h)$ requires a new training stage and a new evaluation stage (on the validation set) for each hyper-parameter configuration h. Thus, depending on the number of hyper-parameters to optimize and on the complexity of the model, which are generally both high in deep learning-based approaches, this process can be expensive. Common hyper-parameter search algorithms, such as grid search and random search, do not take advantage of the results of the previous iterations to select the next hyper-parameters to evaluate (the latter are, thus, uninformed by past evaluations), often wasting computational time on hyper-parameters with poor performance. Conversely, BO methods keep track of results related to past evaluations to update a probabilistic model, mapping hyper-parameters to the probability to obtain a specific score from the objective function. That is, BO can be formalized as a sequential model-based optimization, performing several iterations each one considering a specific hyper-parameter configuration and updating the probability model. This model is a surrogate of the objective function to be minimized and is easier to optimize than the original objective function. BO methods suggest the hyper-parameters in an informed way after each iteration step, based on a "selection function". By investigating hyper-parameters that seem promising based on past results, BO methods can find better configurations than other approaches within fewer iterations compared to uninformed algorithms (e.g., grid or random search) [69].

Hyper-parameter search settings and analysis

Optimal hyper-parameters were searched via BO (using the Python library Optuna [70], version 2.3.0), using tree-structured Parzen estimator as surrogate function and expected improvement as selection function, and sequential model-based optimization was performed for 100 iterations (which is the default value [70]). The hyper-parameters of Sinc-ShallowNet-v2 subjected to BO, defining the array $h \in H$, were ${K_0}^{ISS}$, ${D_1}^{ISS}$, ${c_1}^{ISS}$, $p^{ISS} = p^{FST} = p$,

${K_0}^{FST}$, ${F_0}^{FST}$, ${c_0}^{FC}$ and lr. Within each BO iteration, the hyper-parameters were sampled from the distributions specified in Table 3.4. Note that, based on the adopted distributions, since ${K_0}^{FST}$ can be equal to or less than (but not greater than) ${K_0}^{ISS} \cdot {D_1}^{ISS}$, only compressed or equal (but not overcomplete) representations in FST layer were examined. BO was applied to models trained with WS-WS, WS-CS and LOSO strategies to investigate the optimal design of Sinc-ShallowNet-v2 depending on the training strategy (see Figure 3.2b). BO was not applied to TL-WS, as models trained in this strategy strictly depended on LOSO models. Indeed, in TL-WS the ICNN was initialized with LOSO parameters and then trained on the held-back subject. Therefore, the hyper-parameter configuration adopted in TL-WS models needed to be the same as in the LOSO strategy, i.e., hyper-parameters needed to be kept fixed as the ones of Bayesian-optimized LOSO models. Therefore, also the WS-WS models used for comparison with TL-WS models, while training and evaluating decoders with a progressively increasing size of training and validation sets, were assigned to the Bayesian-optimized hyper-parameters of the LOSO models, for a fair comparison. It is worth remarking that while analyzing the potentialities of transfer learning with TL-WS and WS-WS models, where hyper-parameters were inherited from LOSO models, the validation set was used only to perform early stopping.

Table 3.4 – Searched hyper-parameters of Sinc-ShallowNet-v2: distributions and admitted values sampled during Bayesian optimization. Curly brackets denote discrete admitted values, while square brackets denote interval of admitted values.

Hyper-parameter	Distribution	Values
${K_0}^{ISS}$	uniform	{4, 8, 16}
${D_1}^{ISS}$	uniform	{1, 2, 4}
${c_1}^{ISS}$	uniform	{None, 0.25, 0.5, 0.75, 1, 1.25, 1.5}
$p^{ISS} = p^{FST} = p$	uniform	{None, 0.25, 0.5}
${K_0}^{FST}$	uniform	$\{{K_0}^{ISS} \cdot {D_1}^{ISS}, 1, 2\}$
${F_0}^{FST}$	uniform	{(1,5), (1,9), (1,13), (1,17), (1,21)}
${c_0}^{FC}$	uniform	{None, 0.25, 0.5, 0.75, 1, 1.25, 1.5}
lr	log-uniform	[1e-4, 1e-1]

For each training condition (WS-WS, WS-CS and LOSO), and for a given q-th hyper-parameter $\in h$ ($0 \leq q \leq 7$) the optimal values obtained across the Bayesian-optimized ICNNs were extracted (we had 15·7 ICNNs in WS-WS and 15 ICNNs in WS-CS and LOSO). Then, the probability (P) to obtain a specific hyper-parameter value via BO across ICNNs was computed. This allowed a visualization and a comparison of the optimal model design across training conditions. Moreover, for each training condition the importance score z_q was derived by using the fANOVA hyper-parameter importance evaluation algorithm proposed in [71] that fits a random forest regression model predicting the objective value given a parameter configuration. Importance scores sum up to 1 over the investigated hyper-parameters (i.e., $\sum_q z_q = 1$) and the higher the score of a specific hyper-parameter, the higher its importance.

To provide a fair performance comparison with the other two CNN-based approaches, the EEGNet adaptation used in [50] and Sinc-ShallowNet [52] (see Section 3.2.4), the optimal hyper-parameters of EEGNet and Sinc-ShallowNet were searched using BO for each training condition too. In particular, the hyper-parameters of EEGNet and Sinc-ShallowNet were sampled using the same distributions defined as for Sinc-ShallowNet-v2 (see Table 3.4); of

course, Sinc-ShallowNet lacked the hyper-parameters of the FST block (K_0^{FST} and F_0^{FST}) as this block was absent in this architecture.

3.2.7. Explanation technique: spectral and spatial features analysis

The interpretation of the features learned by Sinc-ShallowNet-v2 incorporating subject-specific knowledge ($\theta = \theta^{(s)}$, in WS-CS) could provide insights about neural signatures related to each specific ASD subject in a data-driven way, enabling between-subject variability investigation in ASD. For this analysis, we retrained the ICNN in WS-CS using the most frequent Bayesian-optimized configuration obtained in WS-WS across all subjects and sessions. By doing so, only one fixed hyper-parameter configuration was used to retrain WS-CS models across all subjects, so that the same number of temporal and spatial filters in the ISS block – whose learned features are the objects of the following analysis – was used across subjects. In this way, we were able to evidence differences/similarities among ASD subjects in terms of the learned features, excluding that these differences may arose from differences in ICNN configurations. The fixed configuration was based on the most frequent WS-WS optimal configuration, as the Bayesian-optimized WS-WS models resulted the lightest and fastest to train (see Figure 3.4 and Table 3.5). It is worth noticing that the WS-CS retrained models achieved comparable performance with the Bayesian-optimized WS-CS models (see Section 3.6.1 of Supplementary Materials).

As described in Section 3.2.3-i, the adopted architecture was designed to provide interpretable parameters in the array ${\theta_{ISS}}^{(s)}$. The ICNN processes input trials by filtering out P300-unrelated spectral and spatial information while preserving only the most significant ones for P300 decoding. However, features may have a different importance on the discrimination, i.e., a band-pass filtering in a peculiar frequency range and a subset of electrodes may be more relevant to distinguish the P300 response. Therefore, the processing of θ_{ISS} should also include an explanation technique, devoted to highlight contributions of the more important features in ASD related to P300, realizing the combination ICNN+ET.

In the following, the proposed algorithm based on ICNN+ET for the investigation of subject-specific spectral and spatial P300 features is described and a schematic representation is reported in Figure 3.3.

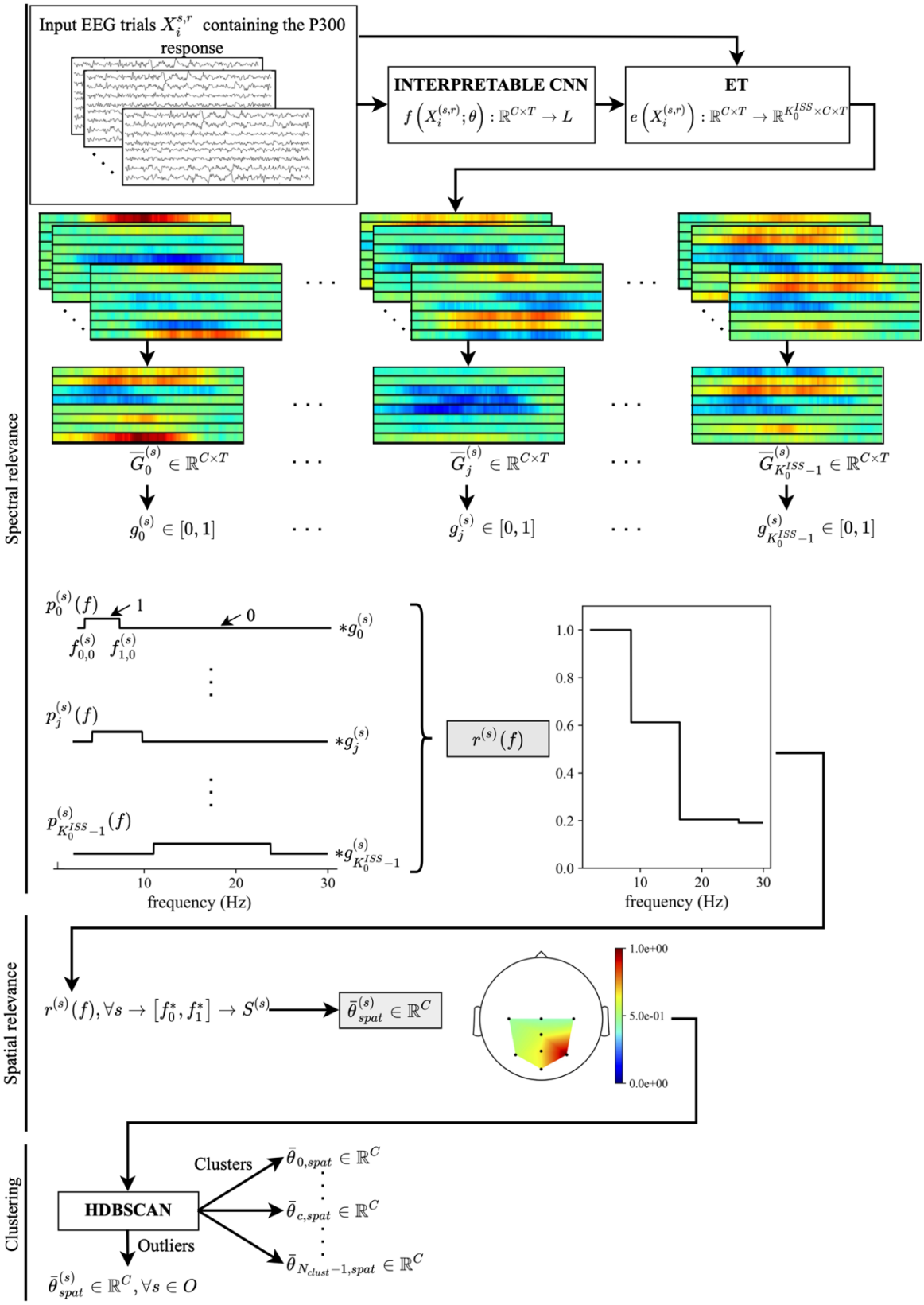


Figure 3.3 – Schematic representation of the algorithm based on ICNN+ET adopted to gain insights about the neural signatures of the visuo-spatial P300 correlate in autism in its three main steps: spectral relevance computation, spatial relevance computation, and clustering.

Computation of the relevance of spectral features (spectral relevance)

At first, given a single input EEG trial of the s-th subject containing the P300 response ($X_i^{(s,r)}$), we computed the saliency [72] for the P300 class of each spatio-temporal sample ($C \cdot T$ samples) within each feature map provided by the temporal sinc-convolutional layer. This ET consists in computing the gradient of the output of the neuron associated to the P300 condition (immediately before the softmax function) with respect to the feature maps provided by the first convolutional layer. The output of the ET is one relevance map for each feature map of the first layer (i.e., $\forall j$, with $0 \leq j \leq {K_0}^{ISS} - 1$), and can be interpreted as a transformation $e\left(X_i^{(s,r)}\right)$: $\mathbb{R}^{C \times T} \rightarrow \mathbb{R}^{{K_0}^{ISS} \times C \times T}$ whose output quantifies how much the spatio-temporal samples in each filtered version of the input affect the P300 prediction. Relevance maps were averaged across P300 trials, and the absolute value of these maps was computed. Therefore, an average relevance map $\bar{G}_j^{(s)} \in \mathbb{R}^{C \times T}, 0 \leq j \leq {K_0}^{ISS} - 1$ was obtained, and finally, the relevance score of each feature map was computed as:

$$g_j^{(s)} = \max_{c,t} \bar{G}_j^{(s)} / \max_{c,t,j} \bar{G}_j^{(s)}, 0 \leq j \leq {K_0}^{ISS} - 1, 0 \leq c \leq C - 1, 0 \leq t \leq T - 1, \quad (3.6)$$

where $g_j^{(s)} \in [0,1]$ is a scalar quantity that summarizes the importance of each band-pass filter for discriminating the P300 response from input EEG trials.

The spectral relevance scores obtained in Equation 3.6 were used to weight the associated passbands of the temporal filters, defined by ${\theta_{spect,j}}^{(s)}$ (see Equation 3.3), computing $r_j^{(s)}(f)$ as follows:

$$\begin{cases} p_j^{(s)}(f) = \begin{cases} 1, \ if \ f_{0,j}^{(s)} \leq f \leq f_{1,j}^{(s)} \\ 0, \ elsewhere \end{cases} & 0 \leq j \leq {K_0}^{ISS} - 1 \\ r_j^{(s)}(f) = g_j^{(s)} \cdot p_j^{(s)}(f) & 0 \leq j \leq {K_0}^{ISS} - 1 \end{cases}. \quad (3.7)$$

In Equation 3.7, $p_j^{(s)}(f)$ indicates the probability that a specific frequency f was included in the passband of the j-th band-pass filter, i.e., marking with a probability of 0 and 1 frequencies outside and inside the passband, respectively.

Then, the spectral relevance $r^{(s)}(f)$, quantifying the relevance of each frequency bin, was obtained as:

$$r^{(s)}(f) = \max_j r_j^{(s)}(f). \quad (3.8)$$

Computation of the relevance of spatial features related to the most relevant spectral features (spatial relevance)

By averaging $r^{(s)}(f)$ across subjects, the frequency ranges retaining a relevance greater than or equal to 0.75 across subjects ($\forall s$) were identified. As described in Section 3.3.3, only one frequency range was identified, and it is indicated in the following with $[f_0^*, f_1^*]$. This interval is characterized by a high relevance with high agreement (≥75%) across subjects, as the spectral relevance was normalized for each subject. For each subject the following analysis was performed. First, we considered the subset of the band-pass filters, denoted as $S^{(s)}$, that contained in their passband the frequency bins belonging to $[f_0^*, f_1^*]$ and we selected the spatial filters (i.e., the learned features $\theta_{jk}^{(s)}$, see Equation 3.4) associated to this subset of band-pass

filters ($j \in S^{(s)}, 0 \leq k \leq D_1^{ISS} - 1$). As we were interested in the investigation of the relevance at the level of single electrode, spatial filters were considered in their absolute value, as done in [45,52]. Subsequently, the absolute spatial features were averaged together electrode per electrode ($\forall c$) and normalized to the maximum across electrodes, obtaining the spatial relevance:

$$\bar{\theta}_{spat}^{(s)} = \frac{1}{N_{spat}^{(s)}} \sum_{j \in S^{(s)},k} abs(\theta_{jk}^{(s)}) \Big/ \max_{c} \frac{1}{N_{spat}^{(s)}} \sum_{j \in S^{(s)},k} abs(\theta_{jk}^{(s)}),\ 0 \leq c \leq C - 1, \qquad (3.9)$$

where $N_{spat}^{(s)}$ represents the overall number of spatial filters associated to the subset $S^{(s)}$ of the band-pass filters (D_1^{ISS} spatial filters for each band-pass filter in the subset).

Based on Equation 3.9, each value of $\bar{\theta}_{spat}^{(s)} \in \mathbb{R}^C$ quantifies the relevance (normalized between [0,1]) of a specific electrode signal filtered in the frequency range most important for decoding the P300 response. Abnormalities in P300 (e.g., ASD abnormalities) may modulate $\bar{\theta}_{spat}^{(s)}$.

Clustering

The spatial relevance $\bar{\theta}_{spat}^{(s)}$ was clustered, automatically finding clusters of subjects characterized by different patterns of $\bar{\theta}_{spat}^{(s)}$. This was made possible by adopting the WS-CS strategy, as it allows the learning of robust subject-specific neural signatures across recording sessions. Here, Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) [73] was used as clustering algorithm (using the Python library hdbscan [74], version 0.8.27), using the Euclidean distance between observations as distance metric. With HDBSCAN, clustering is performed without specifying the number of clusters as input parameter, including also an outlier detection algorithm [73]. Therefore, the adopted clustering algorithm provides the optimal number of clusters N_{clust} populated by the more reliable observations and marks as outliers a subset O of observations. As observations belonging to the same cluster share a common structure, these were averaged together, obtaining $\bar{\theta}_{c,spat}$ for the c-th cluster. Therefore, cluster-level representations were obtained by averaging across a small subset of subjects sharing a peculiar spatial pattern as learned by the model, reflecting neural signatures belonging to different clusters of ASD subjects.

3.2.8. Statistical analysis

Performance

As described in Section 3.2.4, object-level accuracy was used as performance metric, and it was computed separately for each session-specific test set. In addition, as performed in [41], as the performance resulted robust across recording sessions, the performance metric was averaged across all sessions for each subject. Then, the following tests were performed.

i. A Friedmann test was performed to compare the performance obtained by Sinc-ShallowNet-v2 trained with the three training strategies (WS-WS, WS-CS, LOSO strategies). Then, as significant differences were found (see Section 3.3.2), post-hoc

pairwise comparisons were performed testing all combinations; a total of 3 tests were performed to check for differences between different strategies.

ii. For each training strategy, pairwise comparisons were performed between Sinc-ShallowNet-v2 and the other two CNNs that inspired Sinc-ShallowNet-v2 design: the winning solution of the IFMBE 2019 challenge based on EEGNet [50], representing the SOA for P300 decoding, and Sinc-ShallowNet [52]. A total of 6 tests were performed to check for differences between the proposed ICNN and the other two CNN-based solutions.

iii. Pairwise comparisons were performed between Sinc-ShallowNet-v2 trained with TL-WS strategy and trained with WS-WS strategy for each number of calibration blocks used for tuning the models. This was done to test the benefits of pre-training the models on other subjects compared to training models from scratch. A total of 10 tests were performed to check for differences.

All previous pairwise comparisons were performed using Wilcoxon signed-rank tests and false discovery rate correction at $\alpha = 0.05$ using the Benjamini–Hochberg procedure [75] to correct for multiple tests.

ASD clinical scores

Finally, we investigated whether the relevance of spatial features related to the most relevant spectral features of the P300 response, as learned by the ICNN, was related to ASD clinical scores (ADOS, ADI-R, and IQs, see Section 3.2.1). As the cluster analysis highlighted a strong contribution of parietal sites across clusters (see Section 3.3.3), $\bar{\theta}_{spat}^{(s)}$ was averaged across P3, Pz and P4 sites (thus, resulting in a scalar value for each subject s). Then, the Pearson correlation coefficient between this ICNN+ET-derived measure and each ASD clinical score was computed. Lastly, we performed a similar correlation analysis by using a measure derived from subject-specific evoked potentials, to investigate whether useful ASD neural signatures were enhanced by the proposed ICNN+ET analysis or they emerged already from evoked potentials. To this aim, for each subject, EEG trials of the test set containing the P300 response were band-pass filtered in the range $[f_0^*, f_1^*]$ and were averaged together. Then, the resulting evoked potential was averaged across P3, Pz and P4 sites and across the time samples where the P300 response was stronger, i.e., between 400-600 ms, see Borra et al. [43]. Thus, after this averaging procedure, a scalar value was obtained for each subject s. Then, the Pearson correlation coefficient between this ERP-derived measure and each ASD clinical score was computed.

3.3. RESULTS

First, this section describes the results of AHPS via BO and the obtained decoding performance; then, it shows the results obtained via the proposed ICNN+ET algorithm to analyze the neural signatures corresponding to attentional P300 in ASD.

3.3.1. Hyper-parameter search via Bayesian optimization

The hyper-parameter configurations and their importance scores were separately obtained for each tested training strategy (WS-WS, WS-CS, LOSO). Figure 3.4 shows the probability to obtain a specific hyper-parameter value (P) among the admitted ones (Figure 3.4a) and the hyper-parameter importance scores z (Figure 3.4b) for each training condition.

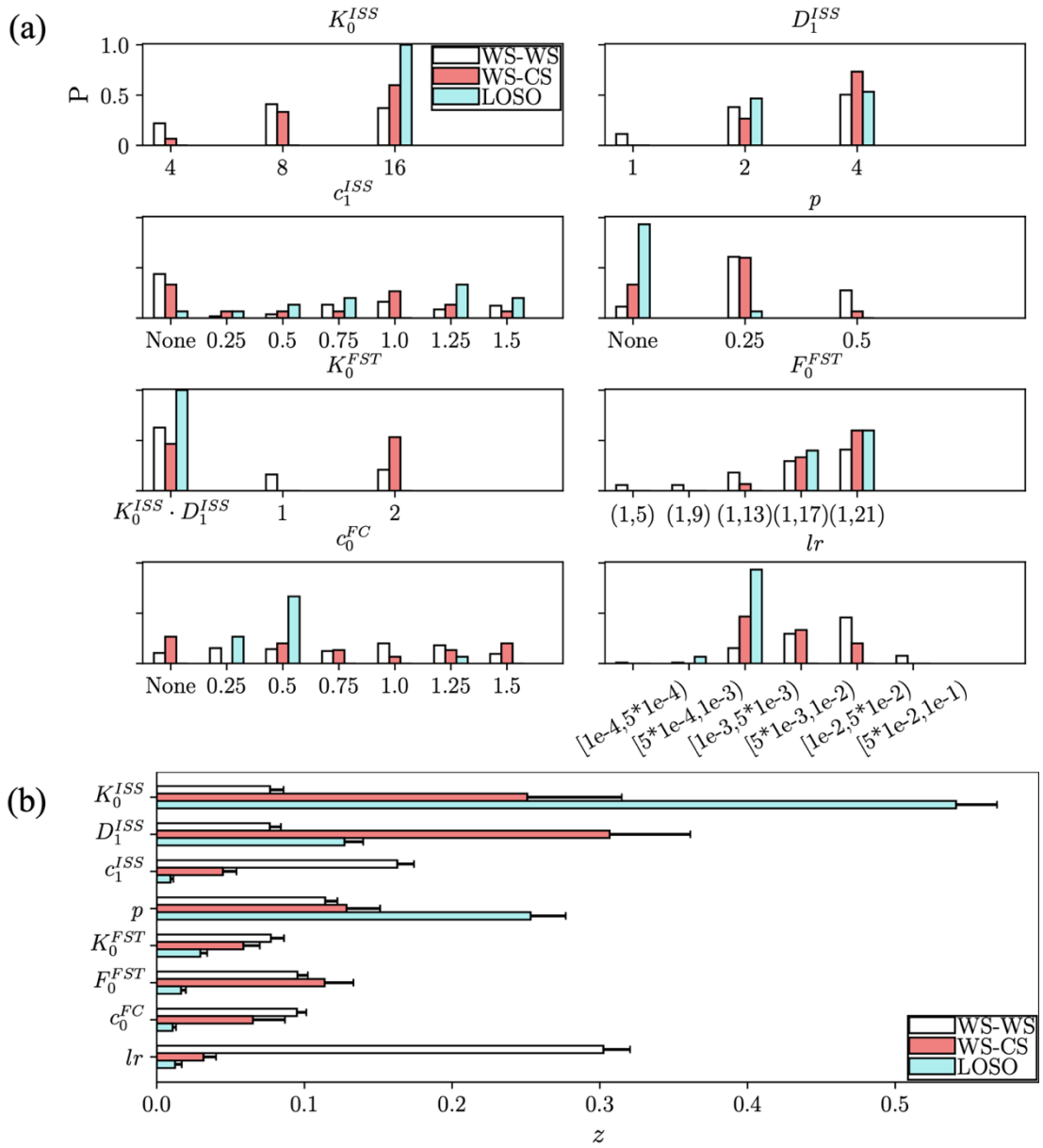


Figure 3.4 – Hyper-parameter distributions (a) and importance scores (b) when training Sinc-ShallowNet-v2 with within-subject and within-session (WS-WS), within-subject and cross-session (WS-CS) and leave-one-subject-out (LOSO) strategies. Hyper-parameter distributions were reported representing the probability (P) that a specific

hyper-parameter value resulted as optimal via BO, and it is reported separately for the different training strategies. Hyper-parameter importance scores were reported in their mean values across decoders trained with each training strategy, separately. Bar heights represent the mean value, while the error bars represent the standard error of the mean.

The more frequent design of the ISS block included a lower number of band-pass filters in WS-WS than WS-CS and LOSO (i.e., ${K_0}^{ISS} = 8$ vs. ${K_0}^{ISS} = 16$) and the same number of spatial filters for each band-pass filter across training conditions (i.e., ${D_1}^{ISS} = 4$). Among the regularization techniques employed, spatial kernel max-norm constraint was applied more frequently in the LOSO strategy, while was progressively applied less frequently (i.e., ${c_1}^{ISS} = None$) moving to WS-CS and WS-WS strategies. The FST block was mostly designed learning an equal representation over the input ISS feature maps (i.e., ${K_0}^{FST} = {K_0}^{ISS} \cdot {D_1}^{ISS}$) for WS-WS and LOSO strategy, while equal representation and compressed representation (with ${K_0}^{FST} = 2$) were almost evenly frequent in WS-CS strategy. Interestingly, the most frequent temporal window in which features were learned in the FST block was approx. of 600 ms for all training strategies ($= {F_0}^{FST}[1]/\,(sf//4)$, where ${F_0}^{FST} = (1,21)$ and $sf = 128$ Hz), and this could be related to the temporal dynamics of the P300 response. Regarding the FC block, weights were always constrained in case of LOSO strategy mainly with low upper bound (c_0^{FC} = 0.25 or 0.5), while the probability distribution of this hyper-parameter appeared almost uniform for the other strategies.

Dropout (both in the ISS and in the FST blocks) was mostly absent in LOSO strategy while it was included in the other strategies with a more frequent dropout probability of $p = 0.25$. Lastly, higher learning rates were adopted in WS-WS strategy, i.e., $lr \in [5 \cdot 1e-3, 1e-2)$, compared to the other strategies where $lr \in [5 \cdot 1e-4, 1e-3)$.

As shown in Figure 3.4b, the learning rate resulted the most important hyper-parameter to optimize in case of WS-WS strategy, while the other hyper-parameters resulted almost equally important. Regarding the WS-CS strategy, the number of band-pass filters (${K_0}^{ISS}$) and of spatial filters for each band-pass filter (${D_1}^{ISS}$) were the most important hyper-parameters, while regarding the LOSO strategy, the number of band-pass filters (${K_0}^{ISS}$) and the dropout probability (p) were the most important ones.

Table 3.5 lists the model size (expressed as the number of trainable parameters) and training time (expressed in s/epoch) of the models. As expected, WS-WS models were the lightest (overall and within each block) and fastest to train, while LOSO models were the heaviest (overall and within each block) and slowest to train. Interestingly, the proportion of trainable parameters across blocks between WS-WS and WS-CS models did not change substantially (ISS: 28%, 28%; FST: 62%, 63%; FC: 10%, 9%, respectively for WS-WS and WS-CS); compared to WS-WS and WS-CS strategies, LOSO models presented a higher concentration of the trainable parameters (i.e., higher capacity) in the FST block (i.e., ISS: 12%; FST: 80%; FC: 8%).

Table 3.5 – Model size (expressed as the number of trainable parameters) and training time (expressed in s/epoch) of Sinc-ShallowNet-v2 for each training condition. The total number of trainable parameters (mean ± standard error of the mean) is reported together with the number of parameters specific for each block, indicating within brackets the percentage of parameters exploited in each block.

	Block	Within-subject and within-session (WS-WS)	Within-subject and cross-session (WS-CS)	Leave-one-subject-out (LOSO)
Model size (# tr. parameters)	-	1207±141	1655±412	4638±559
	ISS	336±21 (28%)	466±47 (28%)	555±43 (12%)
	FST	749±121 (62%)	1042±368 (63%)	3689±483 (80%)
	FC	122±13 (10%)	147±40 (9%)	394±34 (8%)
Training time (s/epoch)	-	0.980±0.014	6.380±0.319	23.1±1.1

3.3.2. Decoding performance

Comparison with EEGNet and Sinc-ShallowNet in different training strategies

The performance metrics averaged across subject-specific and session-specific test sets (see Section 3.2.8) are reported here. Figure 3.5 displays the performance metrics of the Bayesian-optimized EEGNet, the Bayesian-optimized Sinc-ShallowNet, and the Bayesian-optimized Sinc-ShallowNet-v2. Sinc-ShallowNet-v2 scored significant different performance across the three training strategies ($p < 0.001$, Friedmann test). As expected, post-hoc comparisons highlighted that the network performed significantly worse in LOSO strategy than both WS-WS and WS-CS ($p < 0.001$ for both comparisons) strategies. Lastly, the network in WS-CS performed significantly better than WS-WS strategy ($p = 2.3 \cdot 1e - 3$). Sinc-ShallowNet-v2 scored object-level accuracies of 88.5±2.3%, 91.5±1.8%, 76.0±3.2%, compared to 85.5±2.5%, 91.7±1.6%, 76.3±3.2%, of EEGNet, and 76.1±3.9%, 84.0±2.9%, 67.6±3.6%, of Sinc-ShallowNet, respectively in WS-WS, WS-CS and LOSO strategies. Remarkably, despite the reduction in model capacity by re-parametrizing part of the architecture to design an interpretable spectral and spatial feature extractor, Sinc-ShallowNet-v2 achieved comparable performance compared to EEGNet for P300 decoding in WS-CS and LOSO strategies ($p = 0.84$), while significantly outperformed it in the WS-WS strategy ($p = 0.01$). Lastly, it is worth noticing that the updated version of Sinc-ShallowNet-v2 significantly outperformed Sinc-ShallowNet in all training strategies.

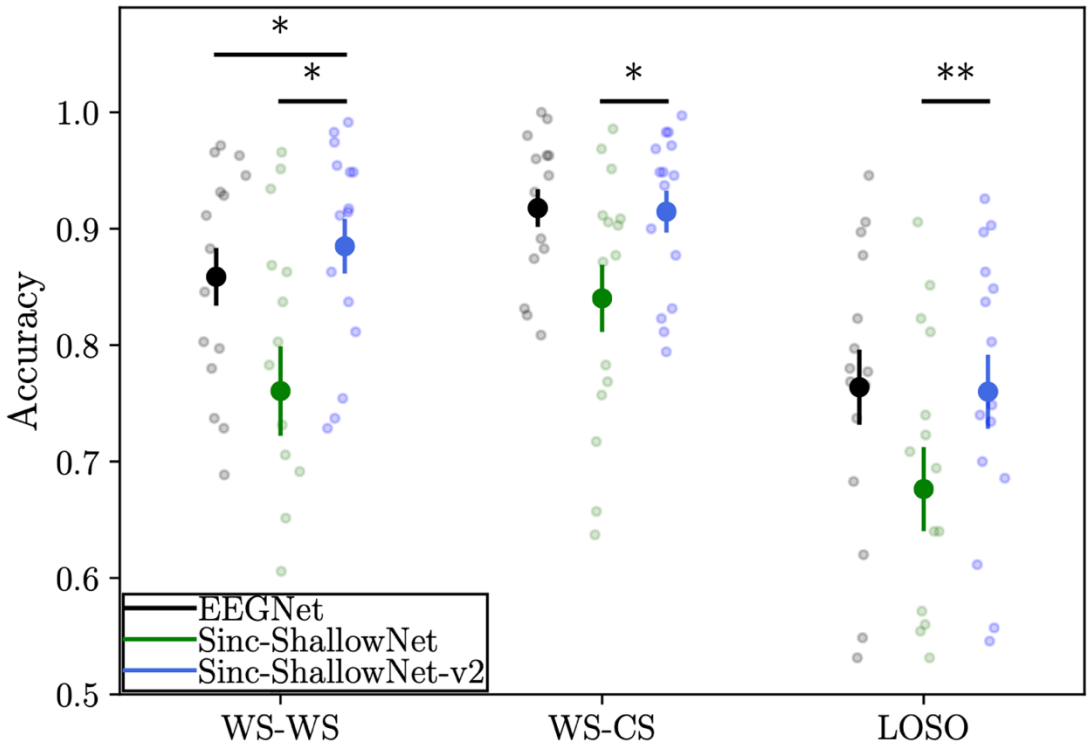


Figure 3.5 – Accuracies of the Bayesian-optimized EEGNet (black), Bayesian-optimized Sinc-ShallowNet (green) and Bayesian-optimized Sinc-ShallowNet-v2 (blue) scored in the WS-WS, WS-CS and LOSO strategies. Bigger dots represent the mean value across subjects, while the error bars represent the standard error of the mean. Smaller dots represent the accuracy scored for each subject (i.e., 15 data points). See part A of Table 3.3 for details about the number of examples defining the training, validation and test sets. Wilcoxon signed-rank tests were performed to compare the performance of the Bayesian-optimized Sinc-ShallowNet-v2 with the Bayesian-optimized EEGNet and with the Bayesian-optimized Sinc-ShallowNet, within each training strategy. P-values were corrected for multiple comparisons (6 in total) via the Benjamini–Hochberg procedure and significant comparisons are marked once applied the correction (*p<0.05, **p<0.01, ***p<0.001). See Section 3.2.8 for further details about the statistical tests.

Transfer learning

Transfer learning was adopted to test the capability of models to transfer the knowledge from other subjects to a new one, for a more practical usage of the decoder in a BCI for ASD treatment (i.e., reduction of BCI calibration times). To this aim, Figure 3.6 shows the accuracy obtained while transferring the knowledge from the other participants to the held-out participant (TL-WS strategy, see Section *Training strategies*-iv), using a progressively increasing number of calibration blocks of the held-out participant (from 2 to 20, where 20 corresponds to the entire subject-specific calibration set, see Section *Training strategies*-i, iv). The WS-WS performance are reported together with TL-WS performance, to highlight the potential benefit of transfer learning compared to a random initialization (proper of the WS-WS strategy). Remarkably, transfer learning was found to be beneficial with significant improvements ($p < 0.01$) for all the tested number of calibration blocks; using just 2 calibration blocks per subject, TL-WS reached an object-level performance as high as 80.3±3.0% with an average improvement as high as 37.1% compared to WS-WS.

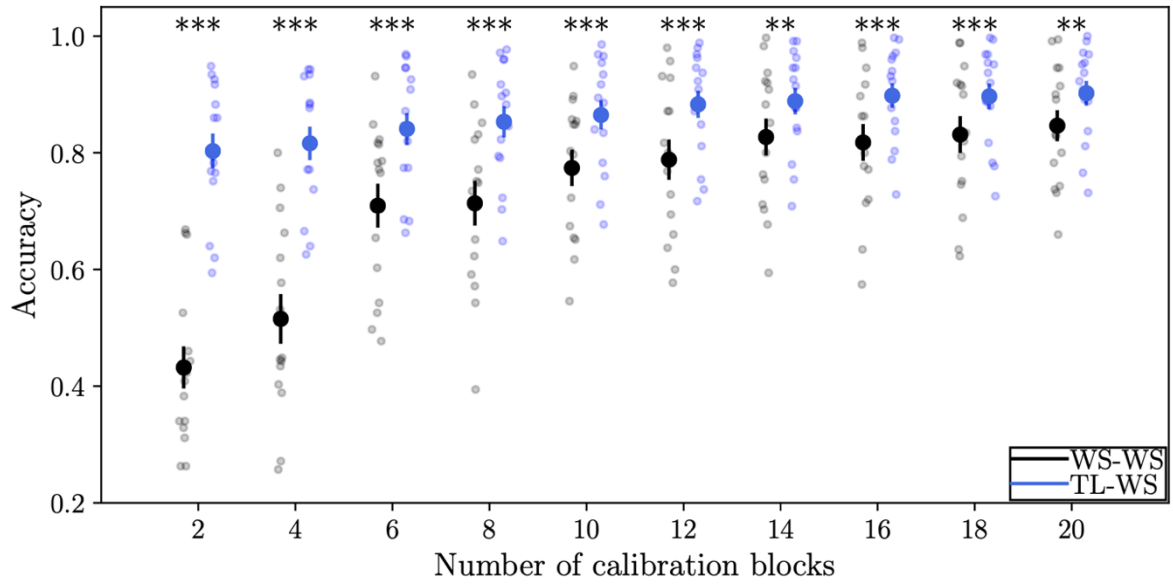


Figure 3.6 – Results of transfer learning as a function of the number of calibration blocks of the subject. Blue distributions show the accuracies obtained with the TL-WS models while transferring the knowledge from multiple subjects on a new subject (i.e., using the knowledge embedded in a LOSO model to initialize the WS-WS model on the held-out subject). Black distributions show the accuracies obtained by randomly initializing the CNN (i.e., WS-WS models trained from scratch, without exploiting knowledge from other subjects). The performance metric is reported for different numbers of calibration blocks used for training the TL-WS and WS models. Bigger dots represent the mean value across subjects, while the error bars represent the standard error of the mean. Smaller dots represent the accuracy scored for each subject (i.e., 15 data points). See part B of Table 3.3 for details about the number of examples defining the training, validation, and test sets. Wilcoxon signed-rank tests were performed to compare the performance of TL-WS models and WS-WS models. P-values were corrected for multiple comparisons via the Benjamini–Hochberg procedure and significant comparisons are marked once applied the correction (*p<0.05, **p<0.01, ***p<0.001). See Section 3.2.8 for further details about the statistical tests.

3.3.3. Subject-specific ASD neural signatures related to P300

Figure 3.7 (top panel) shows the spectral relevance (average ± standard error of the mean across subjects). The frequency range retaining most of the relevance (≥ 0.75) resulted $[f_0^*, f_1^*] = [2, 5.8]$ Hz. Then, by clustering the relevance of spatial features related to this frequency range, two clusters were obtained and their $\bar{\theta}_{c,spat}$ are reported in the bottom panel of Figure 3.7 (left). Each of these cluster-level representations evidenced a peculiar strategy at the level of scalp that the system automatically learned for P300 decoding. Cluster 0 showed a strong and wide contribution of parietal sites (peaking at Pz) symmetrical across hemispheres, while cluster 1 – that resulted the most populated cluster (with 10 observations) – showed a strong right-lateralized contribution at parietal sites (P4). Lastly, in addition to these clusters, two outliers were detected, with a different spatial relevance.

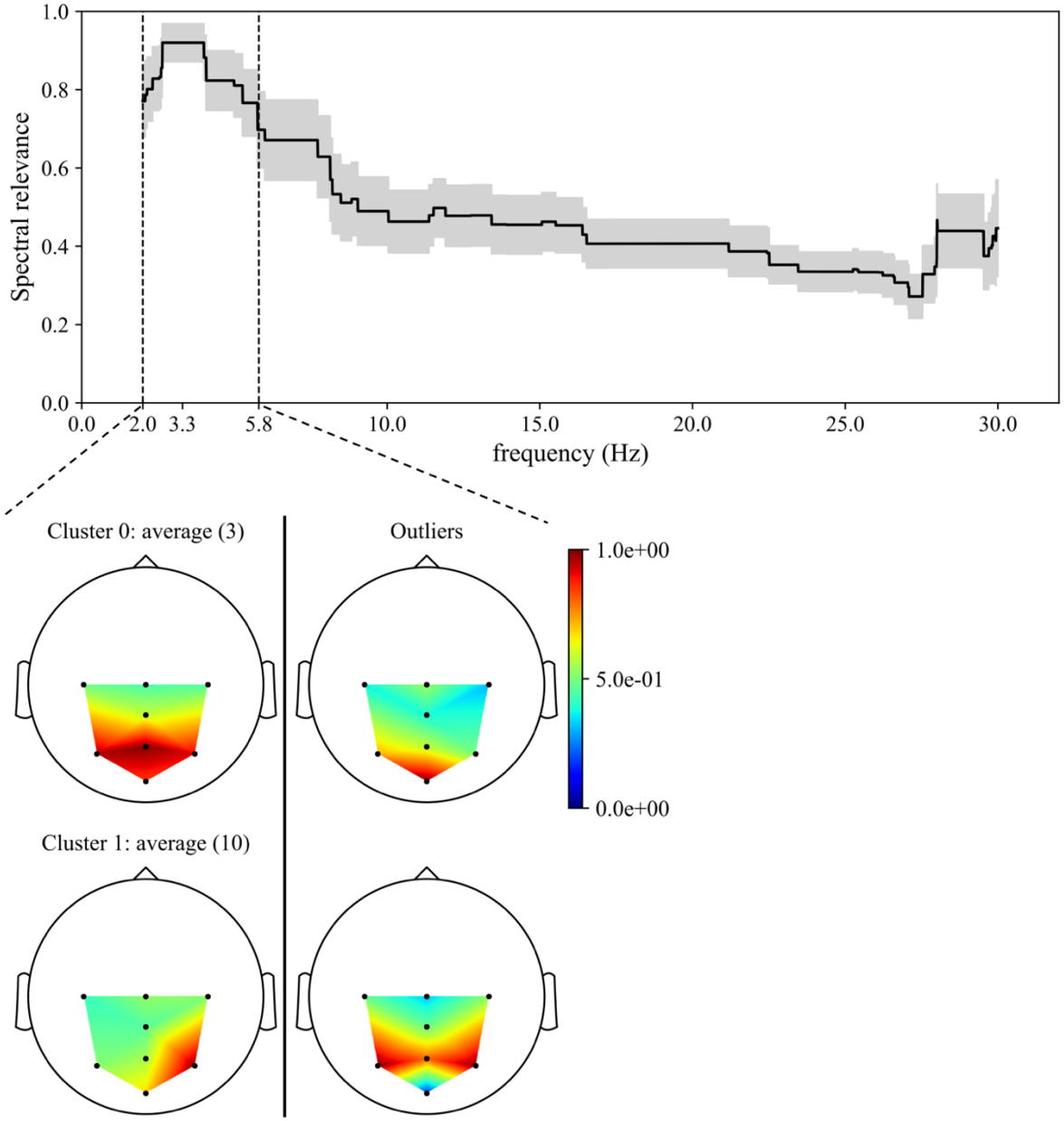


Figure 3.7 – Spectral relevance (top) and spatial relevance (bottom). The spectral relevance is reported in its mean value (black line) and standard error of the mean (grey shaded area) across subjects. The more relevant frequency range (associated to a relevance ≥0.75) is delimited by the two dashed vertical lines. The spatial relevance is displayed for the two clusters (averaged within each cluster) and to the outliers detected. The number of subjects in each cluster is reported within brackets.

The results of the statistical analysis conducted on the relevance of spatial features ($\bar{\theta}_{spat}^{(s)}$) at parietal sites is reported in Figure 3.8. Among all clinical scores, ADOS scores (both A-Communication, B-Social Interaction and A+B-Communication-Social Interaction scores) were the ones that mostly correlated to the ICNN+ET-derived measures, with high (r=0.770, r=0.657 and r=0.801, respectively) and significant ($p < 0.01$) positive correlations. Conversely, by using ERP-derived measures no significant correlations with clinical scores were found.

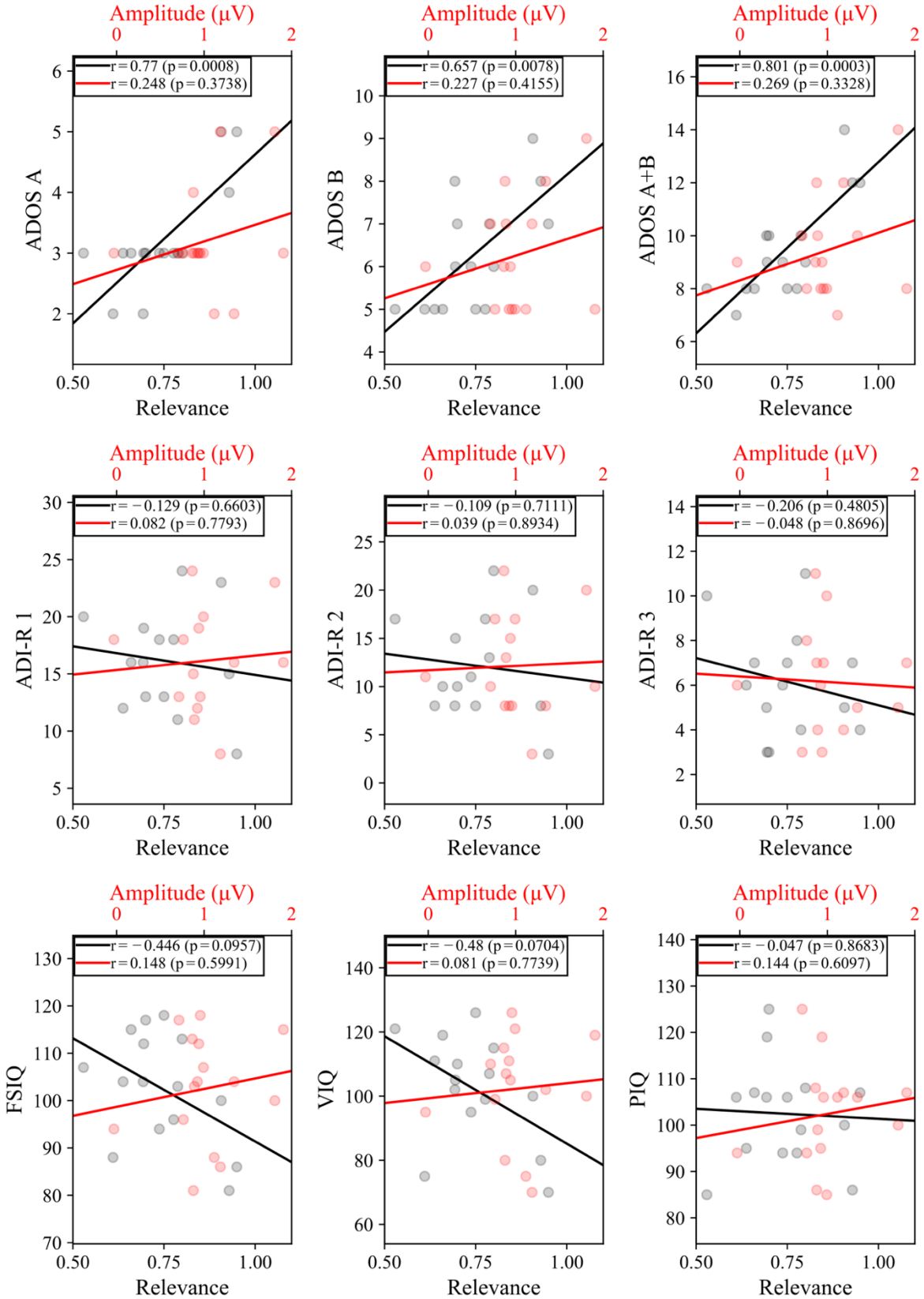


Figure 3.8 – Correlation analysis between each ASD clinical score and the ICNN+ET-derived (black) or ERP-derived measures (red). Subject-specific observations are reported with dots together with regression lines. Pearson's correlation coefficients together with p-values are reported inside each plot.

3.4. DISCUSSION

3.4.1. Hyper-parameter search

Depending on the training strategy adopted, AHPS selected different hyper-parameter configurations for Sinc-ShallowNet-v2. A peculiar trend of the probability distributions related to ${K_0}^{ISS}$, ${D_1}^{ISS}$, ${K_0}^{FST}$, ${F_0}^{FST}$ (see Figure 3.4a) was observed while moving from the WS-WS to WS-CS and LOSO strategies. In particular, the probability distributions of these hyper-parameters progressively moved from distributions more focused on small values (WS-WS) to distributions more focused on higher values (LOSO) among the admitted ones (see Table 3.4 for the admitted values), with the WS-CS strategy exhibiting an intermediate behavior in between the previous strategies. As an example, the probability distributions related to ${K_0}^{ISS}$ moved from a distribution more focused on lower values including also the lowest admitted value (${K_0}^{ISS} = 4$), in the WS-WS strategy, to a distribution entirely focused on the highest admitted value ($P({K_0}^{ISS} = 16) = 1$), in the LOSO strategy. That is, Sinc-ShallowNet-v2, moving from WS-WS to WS-CS and LOSO strategies, required progressively more filtered versions of the input EEG trial (${K_0}^{ISS}$), more electrode combinations tied to each band-pass filtered representation (${D_1}^{ISS}$), more temporal representations to learn (${K_0}^{FST}$) and a wider window size in which learn these representations (${F_0}^{FST}$).

Among regularization techniques, the needing of dropout was stronger in WS-WS strategy, using mostly $p = 0.5$ and $p = 0.25$, with a progressively reduction in WS-CS and LOSO strategies, up to $p = 0$ corresponding to no dropout applied (denoted by $p = None$). That is, the smaller and less variable (in terms of intra-subject and inter-subject variabilities) the dataset, the higher the needing of dropout to provide a better generalization. Similar to other regularization techniques, dropout acts reducing the algorithm capacity. Thus, from the previous considerations about ${K_0}^{ISS}$, ${D_1}^{ISS}$, ${K_0}^{FST}$, ${F_0}^{FST}$ and from the consideration on p, as the training strategy involves a more challenging decoding task, the architecture needs more capacity to solve the task with high performance. This is particularly relevant in strategies such as WS-CS and LOSO, where single-trial EEG decoding is performed using signals collected across several recording sessions (training set with high intra-subject variability) and across several subjects and sessions (training set with high intra-subject and inter-subject variability), respectively. Interestingly, this result was confirmed by looking to the hyper-parameter importance scores (see red and light blue bars in Figure 3.4b), where ${K_0}^{ISS}$ and ${D_1}^{ISS}$, and ${K_0}^{ISS}$ and p were the more important hyper-parameters to optimize for the WS-CS and LOSO strategies, respectively. Furthermore, the need of an increased capacity as observed in the structural hyper-parameters of Sinc-ShallowNet-v2 is reflected onto its model size (see Table 3.5), resulting in 1207, 1655, and 4638 (on average) trainable parameters for WS-WS, WS-CS, and LOSO, respectively. In addition, while progressively increasing the variability in the training examples, the capacity of the model increased differently across the network. While increasing the intra-subject variability (WS-CS) the number of trainable parameters increased by the same proportion across ISS, FST, FC blocks compared to WS-WS, suggesting that increasing the capacity equally across the network could help addressing an increased intra-subject variability in the training examples. Conversely, while also increasing the inter-subject variability (LOSO), the network needed more abstract temporal features to learn in deeper

layers compared to the models trained in the other strategies, i.e., 80% (vs. ~60%) of the parameters were in the FST block (see Table 3.5). This suggests that increasing the capacity at the deeper layers of the architecture could help addressing an increased inter-subject variability.

Lastly, the probability distribution of the learning rate was characterized by being more biased towards higher learning rates in the WS-WS strategy and towards lower learning rates in the LOSO strategy, with the WS-CS representing an intermediate condition. That is, as the decoding task became less challenging, i.e., moving from the LOSO to WS-WS strategies, a higher learning rate resulted beneficial. As for the previous hyper-parameters, this result was confirmed by looking to the importance scores for the WS-WS strategy (Figure 3.4b), where the learning rate was the most important hyper-parameter to search.

3.4.2. Decoding performance

In a preliminary analysis, we investigated the utility of AHPS, by comparing the performance of architectures optimized via BO for each session or each subject instead of using a fixed configuration. Results showed that the use of a fixed configuration provided significantly lower accuracies (see Section 3.6.2 of Supplementary Materials). This suggests that when a new subject or a subject in a new session approaches the BCI, BO should be performed again to achieve higher performance.

Object-level accuracy varied using different training strategies. Despite the larger within-subject variability in the input distributions (due to the involvement of many recording sessions), CNNs trained with WS-CS were able to significantly outperform CNNs trained with WS-WS, although the improvement is modest. A possible explanation for this result may be that some subjects exhibited high variability even intra-session, leading to smaller performance in WS-WS than in WS-CS (subjects 1,13,14 in Supplementary Figure 3.1), on average. Indeed, in these cases WS-CS took great advantage of the larger training set across all sessions, allowing object-level accuracy to be increased up to 8.6% compared to WS-WS. However, WS-WS strategy reached high performance (i.e., >90%) with small variance in most of the subjects (subjects 2, 4, 6, 7, 8, 10, 11, 15 in Supplementary Figure 3.1), suggesting consistency in within-subject responses both intra- and inter-session. In these cases, WS-CS can still benefit from the 7-time increased training set, slightly improving accuracy compared to WS-WS. Lastly, further increasing the decoding difficulty, by including the cross-subject variability into the training distributions, the challenge represented by the LOSO strategy was reflected into the lowest performance across all the training strategies.

Comparison with EEGNet and Sinc-ShallowNet in different training strategies

Sinc-ShallowNet-v2 significantly outperformed the solution based on EEGNet [50] in the WS-WS strategy, while it was comparable in the other strategies (WS-CS and LOSO). Due the introduction of the temporal sinc-convolution in Sinc-ShallowNet-v2, the number of trainable parameters to perform band-pass filtering were only 1.5% of the ones used in EEGNet that exploits conventional temporal convolution. This reduction of trainable parameters provided a beneficial effect when using the training strategy involving the most compact training set (WS-WS), in which a limited algorithm capacity resulted optimal, while no effect was observed in

the conditions with larger and more variable training sets (WS-CS and LOSO), requiring a higher algorithm capacity). Therefore, the introduction of the interpretable spectral and spatial feature extractor, not only resulted beneficial for feature interpretability, but also did not negatively affect the performance. Indeed, surprisingly the performance of Sinc-ShallowNet-v2 was comparable or even significantly higher, while providing at the same time a straightforward access to the spectral and spatial features. Furthermore Sinc-ShallowNet-v2 significantly outperformed the previous version Sinc-ShallowNet [52] in all training conditions. These results suggest that the inclusion of the FST block in Sinc-ShallowNet-v2, by learning temporal features on the output of ISS block, enables the extraction of more relevant P300-related features than the ISS block only, easing the discrimination of the P300 response. In particular, the FST block may better cope with the large variability in time of P300; the lower performance of Sinc-ShallowNet is then ascribable to the lack of this block and to a reduced ability to catch intra-session, inter-session and inter-subject variability.

Transfer learning

Transfer learning could be used (TL-WS strategy) to reduce the calibration time on a new subject recorded in a new recording session (i.e., requiring WS-WS decoding). Indeed, results reported in Figure 3.6 highlight a significant benefit in transferring the knowledge from other participants to a new one recorded in a new session, for each calibration set adopted. Remarkably, the performance improvement was particularly relevant (on average 37.1%) in the lowest data regime (i.e., 2 calibration blocks corresponding to 160 trials), suggesting that the proposed interpretable approach was also capable of transferring the knowledge from other participants enabling its usage with extremely compact-sized calibration sets. Lastly, transfer learning not only improved the performance obtained in a WS-WS strategy using a small portion of calibration blocks, which could be useful in practice to reduce calibration time, but also improved the performance when using all 20 the calibration blocks (i.e., 1600 trials).

3.4.3. Subject-specific ASD neural signatures related to P300

The results obtained by analyzing the neural signatures related to P300 response via the proposed ICNN+ET algorithm suggest that the more relevant (and shared across subjects) features to distinguish P300 were obtained by filtering EEG signals including the frequency range [2, 5.8] Hz in the passband. This is in line with other studies performing P300 classification both on healthy and ASD people, where features from EEG signals were computed mainly in [2,4] Hz, [4,8] Hz, [2,8] Hz and [2,12] Hz frequency ranges [33–35]. In these previous studies, EEG signals recorded from ASD people were pre-processed based on other P300 decoding approaches validated on healthy subjects, adopting fixed cutoff frequencies. Conversely, here the learning system was left free to explore all frequency contents, adapting them to EEG signals of ASD people in an unbiased way. Indeed, some learned filters included also higher cutoff frequencies too, but only the filters containing lower frequency content resulted more important to provide the correct discrimination of the P300 response. Furthermore, the spectral relevance peaked approximatively in [2,4] Hz, matching the findings of the wavelet analysis conducted by Demiralp et al. [36] where single EEG trials were successfully decoded from features designed in delta ([2,4] Hz) frequency range. These

results suggest that future studies on traditional machine learning applications for P300 decoding in ASD, could design filter banks focusing especially on the interval [2, 5.8] Hz, i.e., in the delta and theta ranges.

The spatial relevance showed a strong right-lateralization for most of the observations (see cluster 1 in Figure 3.7). This indicates that the visuo-spatial sensory processing during the BCI task involved mainly the right hemisphere, differing from the classic P300 scalp distribution. In the literature, there is evidence of hemispheric asymmetries underlying social perception [76], e.g., right-lateralization while processing facial expressions related to emotions [77–79]. Furthermore, a right-lateralization was found also in the P300 response in Amaral et al. [80] and was associated to the high-level characteristics in the realism of the animated paradigms provided via the virtual environment (e.g., reflexive attention generated by social gaze orientation). The BCI intervention investigated in this study was based on a visuo-spatial oddball task which exploits a complex animated paradigm (see Section 3.2.1) and here this right-lateralization emerged in the electrode discriminatory power related to P300, as learned by the ICNN. Thus, our results further substantiate the idea that neural signatures related to social perception (e.g., perception of gaze, faces and related gestures) are characterized by a peculiar right-hemispheric asymmetry. Two clusters of spatial relevance for the P300 discrimination were obtained (see Section 3.3.3), potentially highlighting a modulation of the right-hemispheric asymmetry: from an asymmetric involvement of mainly P4 (cluster 1, the most populated – 10 subjects) to the symmetric involvement of mainly Pz, P3 and P4 (cluster 0 – 3 subjects).

From the correlation analysis between the ASD clinical scores and spatial relevance ($\bar{\theta}_{spat}^{(s)}$) of parietal electrodes (P3, Pz, P4), a strong positive and significant correlation with ADOS scores was observed (see Figure 3.8). This result is particularly interesting, as ADOS together with ADI-R are two important assessment tools used to characterize and diagnose ASD [81]. In addition, recent results [82] suggested that ADOS-related scores are more useful than the combination of ADOS and ADI-R scores to diagnose ASD, which is in line with our finding. Notably, at variance with the results obtained with the ICNN+ET-derived measure, no significant correlations were found between ASD clinical scores and the P300 ERP-derived measure.

Overall, we believe that the proposed method presents some points of novelty and strength, compared to previous CNN-based approaches [41,43,45,48–50,57,58], with perspective not only for practical engineering applications but also for theoretical neuroscience knowledge.

First, from a decoding perspective, we optimized our CNN-based decoders in their structure, by tuning their hyper-parameters to achieve higher decoding performance, and this was done separately for different training strategies. To the best of our knowledge, this is the first time that such analysis is performed on P300 CNN-based decoders, using an automatic strategy (Bayesian optimization) and in a training-specific way. Indeed, previous studies mainly proposed CNN structures with hyper-parameters manually selected, without discussing the specific choice [41,45,48–50,57], or performing sensitivity analysis only on few hyper-parameters and for a single training strategy [43,58]. A significant result of our analysis is that for each investigated training strategy (WS-WS, WS-CS, LOSO), different optimal hyper-parameters were found, depending on the variability incorporated into the training examples,

and that a different architecture with a proper capacity should be adopted depending on the specific real-life scenario in which the P300-BCI is used. Thus, the results obtained in this study may help researchers in the design of CNN decoders for P300-based BCI applications, by driving them in the choice of the more appropriate structure that takes into account the adopted training strategy and the level of variability inside the training examples.

Second, we introduced an interpretable layer in our CNN decoders, using convolutional kernels defined as function of only two and directly interpretable parameters per kernel. The performance evaluation of our decoders compared to EEGNet (which is the state-of-the-art for P300 decoding), indicates that this modification, while decreasing the number of trainable parameters, did not reduce the decoding accuracy. Thus, a CNN can be made intrinsically more interpretable without losing decoding capabilities. ICNNs are receiving growing interest in EEG applications, moving beyond the use of CNNs as black-box decoders only, towards a neurophysiological interpretation of the learned features. In this regard, it is highly important to fist verify the decoding performance of ICNNs, to ensure reliability of the learned features as discriminative of the process under investigation, otherwise the neurophysiological interpretation of these features would be unreliable. So far, ICNNs have been adopted in literature to decode and analyze motor states [52,53] and not the P300 response and not even in neurological disorder conditions such as ASD. Thus, the present study serves also for validating ICNNs for P300 decoding in ASD subjects.

Third and related to previous point, by taking advantage of the interpretability embedded into the proposed ICNN, we have formalized an analysis workflow that combines our ICNN with an explanation technique (ET) to derive novel ASD biomarkers related to P300 by exploiting the knowledge learned by the ICNN. The ICNN+ET combination was able to capture and enhance, better than a canonical ERP analysis, meaningful spectral and spatial characteristics underlying visuo-spatial sensory and attentional processing that are related to autism. The enhancement of P300-related features is in line with the results of our recent study [57] on healthy subjects, where representations derived from a CNN+ET framework better highlighted P300 subcomponents (i.e., P3a and P3b) from single trial level, compared to ERP canonical analysis. The remarkable aspect is that not only the ICNN-derived ASD markers obtained here have a straightforward neurophysiological interpretation, but they also appeared more sensitive than markers derived from traditional ERP-analysis.

Of course, the present study has also some limitations that can be the subject of future improvements, and can foster further investigations along this line of research.

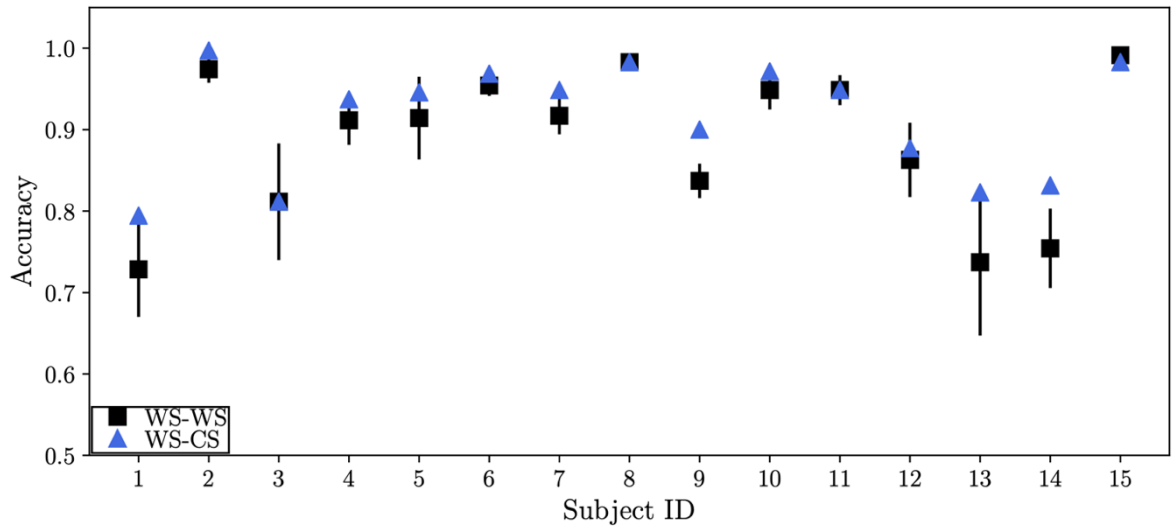
First, the optimal models obtained via Bayesian optimization may be task-related i.e., may result optimal only for EEG decoding of P300 response elicited through the specific kind of stimuli applied during the adopted oddball paradigm (visual stimuli presented inside a virtual environment). It would be of great value to test these same models on other BCI-based P300 paradigms, i.e., using a different type of visual stimuli or a different sensory modality (acoustic stimuli), or even on BCI paradigms based on event-related potentials other than P300. This would test the ability of the proposed models to generalize across different BCI paradigms. This kind of analysis is very important to promote a more efficient use of decoders, limiting the proliferation of novel decoders for every new paradigm and task, but it is only rarely performed (see [52] as a useful example of this analysis).

Another limitation concerns the limited number of subjects included in the examined dataset. Despite the high number of trials recorded for each subject (>30000 across 7 sessions), the analyzed signals were recorded only from 15 subjects. The need for a larger dataset is even more important due to the high heterogeneity in ASD involving a wide spectrum of symptoms. Thus, the biomarkers derived from the ICNN+ET analysis that were found to be correlated with ASD social impairment scores, although promising, require a more extensive validation on a larger set of participants. Furthermore, in future research, the workflow ICNN+ET, applied here to study P300 in autism, could be focused on analyzing other ERP components in autism, and/or in other neurological and neurodevelopmental disorders that involve ERP abnormalities (e.g., schizophrenia, depression, etc.) [83].

3.5. CONCLUSIONS

In conclusion, in this study we investigated the Bayesian-optimized design of an ICNN in different training strategies when decoding the P300 response for BCI intervention in ASD. While performing AHPS, models were found needing more capacity and lower learning rates to decode EEG signals when more variability was included in the training distribution (i.e., including progressively intra-subject and inter-subject variabilities), i.e., depending on the practical scenario in which the decoder is used in the BCI system. Despite its interpretable and lightweight nature, the proposed ICNN performed as well as EEGNet, even significantly outperforming it in the strategy with the lowest variability in the training examples (within-subject and within-session). Furthermore, Sinc-ShallowNet-v2 significantly outperformed the previous Sinc-ShallowNet design. Lastly, transferring the knowledge from other users to a new one proved to lead to a substantial reduction of calibration times. All these results contribute to the development of optimal decoders for P300-based BCI for ASD interventions, by specifically improving CNN designs, performance, and calibration times. Furthermore, in this study we leveraged the interpretable nature embedded into the ICNN to design an ICNN+ET algorithm for the analysis of the visuo-spatial P300 in ASD in the frequency and spatial domains. The analysis on spectral features matched known P300-related correlates, and while analyzing spatial features, a right-hemispheric asymmetry was found, in line with the literature of social perception. The modulation of this asymmetry, as provided by the ICNN+ET analysis, was found to be correlated to ADOS scores, while no significant correlations with any clinical score were found by using a simpler ERP analysis. This suggests that the ICNN+ET algorithm was capable of better characterize and enhance useful ASD-related features than a canonical analysis. In the future, the analysis on optimal ICNN designs for P300 decoding may be extended to other oddball recording paradigms involving different stimuli properties. Furthermore, the proposed ICNN+ET combination could be applied on more subjects for further validation, generalized to other ERP components and neurological disorders to study alterations in ERP components in a data-driven way, and also possibly extended on other recording modalities of neural activity, e.g., magnetoencephalography.

3.6. SUPPLEMENTARY MATERIALS



Supplementary Figure 3.1 – Accuracies scored for each subject with the Bayesian-optimized Sinc-ShallowNet-v2 trained with the within-subject and within-session (WS-WS) strategy – black – and with the within-subject and cross-session (WS-CS) – blue. Markers (squares and triangles) represent the mean value across sessions, while the error bars represent the standard error of the mean.

3.6.1. ICNN hyper-parameter configuration used in WS-CS models to analyze P300 spectral and spatial features in autism

To perform the analysis on the spectral and spatial features related to P300 in ASD (see Section 3.2.7) by processing the same number of spectral and spatial features across subjects (i.e., the same number of temporal and spatial filters in the ISS block), the ICNN was retrained in WS-CS using one fixed hyper-parameter configuration. This configuration was defined as the most frequent Bayesian-optimized configuration obtained in WS-WS across all subjects and sessions, as the Bayesian-optimized WS-WS models resulted the lightest and fastest to train (see Figure 3.4 and Table 3.5). Using this fixed configuration for all subjects and sessions, Sinc-ShallowNet-v2 scored an accuracy of 90.7±2.0% (mean value ± standard error of the mean) vs. 91.5±1.8% scored by the Bayesian-optimized Sinc-ShallowNet-v2 (see Section 3.3.2). Performing a pairwise comparison with a Wilcoxon signed-rank test, the retrained models with a fixed configuration resulted comparable ($p > 0.05$) in terms of performance respect to the Bayesian-optimized models.

3.6.2. Performance of WS-WS models with fixed ICNN hyper-parameter configurations derived from previous results of Bayesian optimization in BCI scenarios

In this section, we evaluated whether the application of Bayesian optimization in each WS-WS model was beneficial compared to using a fixed hyper-parameters configuration across different sessions and subjects, addressing two practical scenarios where a new subject or a subject in a new session approaches the BCI:

i. For each subject and each session, the ICNN hyper-parameter configuration was set using the most frequent configuration obtained in BO with the WS-WS strategy resulting from all other subjects' sessions. This condition was devoted to analyzing the effect on

performance of using a WS-WS design that resulted optimal for other subjects' sessions, and thus, to evaluate whether BO is beneficial for a new subject compared to a fixed configuration resulting from previously performed BO on other subjects.

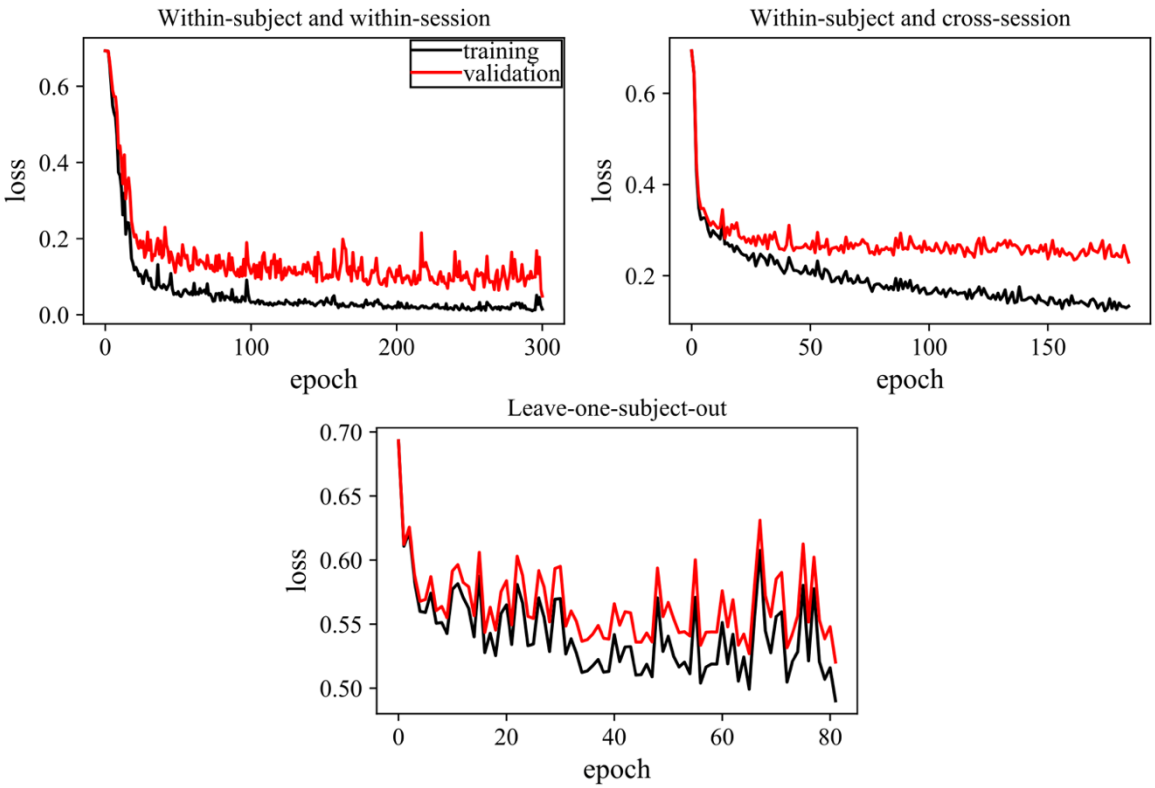
ii. For each subject and each session, the ICNN hyper-parameter configuration was set using the most frequent configuration obtained in BO with the WS-WS strategy from all other sessions of that specific subject. This condition was devoted to analyzing the impact on performance when using a WS-WS design that resulted optimal for other sessions of the subject and thus, to evaluate whether BO is beneficial for a new session compared to a fixed configuration resulting from previously performed BO on other sessions of that subject.

It is worth remarking that the fixed configuration adopted here in points i. and ii. differ from the fixed configuration adopted in Section 3.6.1 of Supplementary Materials, where all subjects and sessions, including the target subject or session the ICNN was trained on, were considered when computing the most frequent hyper-parameter configuration. Conversely, in this section the most frequent hyper-parameter configuration is computed in a subject- or session-agnostic way, i.e., without exploiting any knowledge about the target subject or session the ICNN was trained on. This was done to study practical scenarios where a new subject or session is recorded.

Exploiting Bayesian-optimized configurations obtained from other subjects (point i.) or from other sessions of the subject separately (point ii.) to set a fixed ICNN configuration, accuracies of 84.0±2.9% (mean value ± standard error of the mean) and 82.3±3.9% were obtained, respectively for points i. and ii. Pairwise comparisons (Wilcoxon signed-rank tests corrected for multiple tests with false discovery rate as described in Section 3.2.8) between the ICNNs defined with the fixed configurations as in point i. and ii. vs. the Bayesian-optimized ICNNs trained in WS-WS (88.5±2.3%, see Section 3.3.2), indicated that a significant worsening in performance was obtained ($p < 0.001$ and $p < 0.01$, respectively) when using fixed configurations. Therefore, when a new subject (as modelled in point i.) or a subject in new session (as modelled in point ii.) approaches the BCI, BO should be performed again to achieve higher performance.

3.6.3. Representative example of training and validation losses during training

In Supplementary Figure 3.2, the training and validation losses for a representative subject (WS-CS and LOSO strategies) and session (WS-WS strategy) are reported as a function of the training epochs, for each training strategy.



Supplementary Figure 3.2 – Training (black) and validation (red) losses of Sinc-ShallowNet-v2 using different training strategies: within-subject and within-session (WS-WS), within-subject and cross-session (WS-CS), leave-one-subject-out (LOSO). The displayed curves refer to an ICNN trained on the same representative subject (WS-CS and LOSO); in case of WS-WS a representative session of the same subject was considered.

From Supplementary Figure 3.2, similar loss values were obtained while using WS-WS and WS-CS strategies. On the contrary, in comparison with WS-WS and WS-CS, training and validation losses during the LOSO optimization exhibited more fluctuations as the training proceeded and LOSO optimization ended at higher values of both training and validation losses. This behavior can be attributed to the inherently higher difficulty of LOSO training, due to the challenging task of capturing P300 relevant features from training examples with high variability, including both inter-session and inter-subject variability. This is also reflected into the LOSO performance, which resulted the lowest across the different training strategies (see Section 3.4.2). However, despite higher losses and expected poorer performance, LOSO models provided better initialization points when transferring the knowledge from other subjects to a new one (i.e., when performing transfer learning), as discussed in Section 3.4.2.

3.7. REFERENCES

[1] American Psychiatric Association 2013 *Diagnostic and Statistical Manual of Mental Disorders* (American Psychiatric Association)

[2] Baron-Cohen S 1989 Perceptual role taking and protodeclarative pointing in autism *British Journal of Developmental Psychology* **7** 113–27

[3] Baron-Cohen S, Baldwin D A and Crowson M 1997 Do Children with Autism Use the Speaker's Direction of Gaze Strategy to Crack the Code of Language? *Child Development* **68** 48

[4] Swettenham J, Baron-Cohen S, Charman T, Cox A, Baird G, Drew A, Rees L and Wheelwright S 1998 The Frequency and Distribution of Spontaneous Attention Shifts between Social and Nonsocial Stimuli in Autistic, Typically Developing, and Nonautistic Developmentally Delayed Infants *Journal of Child Psychology and Psychiatry* **39** 747–53

[5] Klin A, Jones W, Schultz R, Volkmar F and Cohen D 2002 Visual Fixation Patterns During Viewing of Naturalistic Social Situations as Predictors of Social Competence in Individuals With Autism *Arch Gen Psychiatry* **59** 809

[6] Dawson G, Toth K, Abbott R, Osterling J, Munson J, Estes A and Liaw J 2004 Early Social Attention Impairments in Autism: Social Orienting, Joint Attention, and Attention to Distress. *Developmental Psychology* **40** 271–83

[7] Bakeman R and Adamson L B 1984 Coordinating Attention to People and Objects in Mother-Infant and Peer-Infant Interaction *Child Development* **55** 1278

[8] Charman T 1998 Specifying the Nature and Course of the Joint Attention Impairment in Autism in the Preschool Years: Implications for Diagnosis and Intervention *Autism* **2** 61–79

[9] Charman T 2003 Why is joint attention a pivotal skill in autism? ed U Frith and E L Hill *Phil. Trans. R. Soc. Lond. B* **358** 315–24

[10] Travers B G, Adluru N, Ennis C, Tromp D P M, Destiche D, Doran S, Bigler E D, Lange N, Lainhart J E and Alexander A L 2012 Diffusion Tensor Imaging in Autism Spectrum Disorder: A Review: Diffusion tensor imaging *Autism Res* **5** 289–313

[11] Minshew N J and Keller T A 2010 The nature of brain dysfunction in autism: functional brain imaging studies *Current Opinion in Neurology* **23** 124–30

[12] Townsend J, Westerfield M, Leaver E, Makeig S, Jung T-P, Pierce K and Courchesne E 2001 Event-related brain response abnormalities in autism: evidence for impaired cerebello-frontal spatial attention networks *Cognitive Brain Research* **11** 127–45

[13] Luck S J, Hillyard S A, Mouloua M, Woldorff M G, Clark V P and Hawkins H L 1994 Effects of spatial cuing on luminance detectability: Psychophysical and electrophysiological evidence for early selection. *Journal of Experimental Psychology: Human Perception and Performance* **20** 887–904

[14] Heinze H J, Luck S J, Mangun G R and Hillyard S A 1990 Visual event-related potentials index focused attention within bilateral stimulus arrays. I. Evidence for early selection *Electroencephalography and Clinical Neurophysiology* **75** 511–27

[15] Herrmann C S and Knight R T 2001 Mechanisms of human attention: event-related potentials and oscillations *Neuroscience & Biobehavioral Reviews* **25** 465–76

[16] Sokhadze E, Baruth J, Tasman A, Sears L, Mathai G, El-Baz A and Casanova M F 2009 Event-related Potential Study of Novelty Processing Abnormalities in Autism *Appl Psychophysiol Biofeedback* **34** 37–51

[17] Polich J 2007 Updating P300: An integrative theory of P3a and P3b *Clinical Neurophysiology* **118** 2128–48

[18] Azizian A and Polich J 2007 Evidence for Attentional Gradient in the Serial Position Memory Curve from Event-related Potentials *Journal of Cognitive Neuroscience* **19** 2071–81

[19] Mecklinger A and Pfeifer E 1996 Event-related potentials reveal topographical and temporal distinct neuronal activation patterns for spatial and object working memory *Cognitive Brain Research* **4** 211–24
[20] Cui T, Wang P P, Liu S and Zhang X 2017 P300 amplitude and latency in autism spectrum disorder: a meta-analysis *Eur Child Adolesc Psychiatry* **26** 177–90
[21] Ciesielski K T, Courchesne E and Elmasian R 1990 Effects of focused selective attention tasks on event-related potentials in autistic and normal individuals *Electroencephalography and Clinical Neurophysiology* **75** 207–20
[22] Courchesne E, Kilman B A, Galambos R and Lincoln A J 1984 Autism: Processing of novel auditory information assessed by event-related brain potentials *Electroencephalography and Clinical Neurophysiology/Evoked Potentials Section* **59** 238–48
[23] Courchesne E, Lincoln A J, Kilman B A and Galambos R 1985 Event-related brain potential correlates of the processing of novel visual and auditory information in autism *J Autism Dev Disord* **15** 55–76
[24] Courchesne E, Lincoln A J, Yeung-Courchesne R, Elmasian R and Grillon C 1989 Pathophysiologic findings in nonretarded autism and receptive developmental language disorder *J Autism Dev Disord* **19** 1–17
[25] Dawson G, Finley C, Phillips S, Galpert L and Lewy A 1988 Reduced P3 amplitude of the event-related brain potential: Its relationship to language ability in austism *J Autism Dev Disord* **18** 493–504
[26] Verbaten M N, Roelofs J W, van Engeland H, Kenemans J K and Slangen J L 1991 Abnormal visual event-related potentials of autistic children *J Autism Dev Disord* **21** 449–70
[27] Kemner C, van der Gaag R J, Verbaten M and van Engeland H 1999 ERP differences among subtypes of pervasive developmental disorders *Biological Psychiatry* **46** 781–9
[28] Amaral C, Mouga S, Simões M, Pereira H C, Bernardino I, Quental H, Playle R, McNamara R, Oliveira G and Castelo-Branco M 2018 A Feasibility Clinical Trial to Improve Social Attention in Autistic Spectrum Disorder (ASD) Using a Brain Computer Interface *Frontiers in Neuroscience* **12** 477
[29] Amaral C P, Simões M A, Mouga S, Andrade J and Castelo-Branco M 2017 A novel Brain Computer Interface for classification of social joint attention in autism and comparison of 3 experimental setups: A feasibility study *Journal of Neuroscience Methods* **290** 105–15
[30] Yger F, Berar M and Lotte F 2017 Riemannian Approaches in Brain-Computer Interfaces: A Review *IEEE Trans. Neural Syst. Rehabil. Eng.* **25** 1753–62
[31] Saha S and Baumert M 2020 Intra- and Inter-subject Variability in EEG-Based Sensorimotor Brain Computer Interface: A Review *Front. Comput. Neurosci.* **13** 87
[32] Lotte F, Bougrain L, Cichocki A, Clerc M, Congedo M, Rakotomamonjy A and Yger F 2018 A review of classification algorithms for EEG-based brain–computer interfaces: a 10 year update *Journal of Neural Engineering* **15** 031005
[33] de Arancibia L, Sánchez-González P, Gómez E J, Hernando M E and Oropesa I 2020 Linear vs Nonlinear Classification of Social Joint Attention in Autism Using VR P300-Based Brain Computer Interfaces *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1869–74
[34] Krzemiński D, Michelmann S, Treder M and Santamaria L 2020 Classification of P300 Component Using a Riemannian Ensemble Approach *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1885–9

[35] Chatterjee B, Palaniappan R and Gupta C N 2020 Performance Evaluation of Manifold Algorithms on a P300 Paradigm Based Online BCI Dataset *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* IFMBE Proceedings vol 76, ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1894–8
[36] Demiralp T, Ademoglu A, Istefanopulos Y, Başar-Eroglu C and Başar E 2001 Wavelet analysis of oddball P300 *International Journal of Psychophysiology* **39** 221–7
[37] Bostanov V and Kotchoubey B 2006 The t-CWT: A new ERP detection and quantification method based on the continuous wavelet transform and Student's t-statistics *Clinical Neurophysiology* **117** 2627–44
[38] Bittencourt-Villalpando M and Maurits N M 2018 Stimuli and Feature Extraction Algorithms for Brain-Computer Interfaces: A Systematic Comparison *IEEE Trans. Neural Syst. Rehabil. Eng.* **26** 1669–79
[39] Bittencourt-Villalpando M and Maurits N M 2020 Linear SVM Algorithm Optimization for an EEG-Based Brain-Computer Interface Used by High Functioning Autism Spectrum Disorder Participants *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1875–84
[40] Miladinović A, Ajčević M, Battaglini P P, Silveri G, Ciacchi G, Morra G, Jarmolowska J and Accardo A 2020 Slow Cortical Potential BCI Classification Using Sparse Variational Bayesian Logistic Regression with Automatic Relevance Determination *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* IFMBE Proceedings vol 76, ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1853–60
[41] Simões M, Borra D, Santamaría-Vázquez E, GBT-UPM, Bittencourt-Villalpando M, Krzemiński D, Miladinović A, Neural_Engineering_Group, Schmid T, Zhao H, Amaral C, Direito B, Henriques J, Carvalho P and Castelo-Branco M 2020 BCIAUT-P300: A Multi-Session and Multi-Subject Benchmark Dataset on Autism for P300-Based Brain-Computer-Interfaces *Front. Neurosci.* **14** 568104
[42] Craik A, He Y and Contreras-Vidal J L 2019 Deep learning for electroencephalogram (EEG) classification tasks: a review *J. Neural Eng.* **16** 031001
[43] Borra D, Fantozzi S and Magosso E 2021 A Lightweight Multi-Scale Convolutional Neural Network for P300 Decoding: Analysis of Training Strategies and Uncovering of Network Decision *Frontiers in Human Neuroscience*
[44] Lindsay G 2020 Convolutional Neural Networks as a Model of the Visual System: Past, Present, and Future *Journal of Cognitive Neuroscience* 1–15
[45] Cecotti H and Graser A 2011 Convolutional Neural Networks for P300 Detection with Application to Brain-Computer Interfaces *IEEE Transactions on Pattern Analysis and Machine Intelligence* **33** 433–45
[46] Srivastava N, Hinton G, Krizhevsky A, Sutskever I and Salakhutdinov R 2014 Dropout: a simple way to prevent neural networks from overfitting *The Journal of Machine Learning Research* **15** 1929–58
[47] Ioffe S and Szegedy C 2015 Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift *Proceedings of the 32nd International Conference on Machine Learning* Proceedings of Machine Learning Research vol 37, ed F Bach and D Blei (Lille, France: PMLR) pp 448–56
[48] Manor R and Geva A B 2015 Convolutional Neural Network for Multi-Category Rapid Serial Visual Presentation BCI *Frontiers in Computational Neuroscience* **9** 146

[49] Lawhern V J, Solon A J, Waytowich N R, Gordon S M, Hung C P and Lance B J 2018 EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces *Journal of Neural Engineering* **15** 056013
[50] Borra D, Fantozzi S and Magosso E 2020 Convolutional Neural Network for a P300 Brain-Computer Interface to Improve Social Attention in Autistic Spectrum Disorder *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1837–43
[51] Zhang X, Yao L, Wang X, Monaghan J, McAlpine D and Zhang Y 2021 A survey on deep learning-based non-invasive brain signals: recent advances and new frontiers *J. Neural Eng.* **18** 031002
[52] Borra D, Fantozzi S and Magosso E 2020 Interpretable and lightweight convolutional neural network for EEG decoding: Application to movement execution and imagination *Neural Networks* S0893608020302021
[53] Zhao D, Tang F, Si B and Feng X 2019 Learning joint space–time–frequency features for EEG decoding on small labeled data *Neural Networks* **114** 67–77
[54] Liang S, Hang W, Yin M, Shen H, Wang Q, Qin J, Choi K-S and Zhang Y 2022 Deep EEG feature learning via stacking common spatial pattern and support matrix machine *Biomedical Signal Processing and Control* **74** 103531
[55] Borra D, Fantozzi S and Magosso E 2020 EEG Motor Execution Decoding via Interpretable Sinc-Convolutional Neural Networks *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1113–22
[56] Vahid A, Mückschel M, Stober S, Stock A-K and Beste C 2020 Applying deep learning to single-trial EEG data provides evidence for complementary theories on action control *Commun Biol* **3** 112
[57] Borra D and Magosso E 2021 Deep learning-based EEG analysis: investigating P3 ERP components *Journal of Integrative Neuroscience* **In press**
[58] Farahat A, Reichert C, Sweeney-Reed C and Hinrichs H 2019 Convolutional neural networks for decoding of covert attention focus and saliency maps for EEG feature visualization *Journal of Neural Engineering*
[59] Wu W, Zhang Y, Jiang J, Lucas M V, Fonzo G A, Rolle C E, Cooper C, Chin-Fatt C, Krepel N, Cornelssen C A, Wright R, Toll R T, Trivedi H M, Monuszko K, Caudle T L, Sarhadi K, Jha M K, Trombello J M, Deckersbach T, Adams P, McGrath P J, Weissman M M, Fava M, Pizzagalli D A, Arns M, Trivedi M H and Etkin A 2020 An electroencephalographic signature predicts antidepressant response in major depression *Nat Biotechnol* **38** 439–47
[60] Lord C, Risi S, Lambrecht L, Cook, Jr. E H, Leventhal B L, DiLavore P C, Pickles A and Rutter M 2000 The Autism Diagnostic Observation Schedule—Generic: A Standard Measure of Social and Communication Deficits Associated with the Spectrum of Autism *Journal of Autism and Developmental Disorders* **30** 205–23
[61] Lord C, Rutter M and Le Couteur A 1994 Autism Diagnostic Interview-Revised: A revised version of a diagnostic interview for caregivers of individuals with possible pervasive developmental disorders *J Autism Dev Disord* **24** 659–85
[62] Spek A A, Scholte E M and van Berckelaer-Onnes I A 2008 Brief Report: The Use of WAIS-III in Adults with HFA and Asperger Syndrome *J Autism Dev Disord* **38** 782–7
[63] Schirrmeister R T, Springenberg J T, Fiederer L D J, Glasstetter M, Eggensperger K, Tangermann M, Hutter F, Burgard W and Ball T 2017 Deep learning with convolutional neural networks for EEG decoding and visualization *Human brain mapping* **38** 5391–420

[64] Ravanelli M and Bengio Y 2018 Speaker Recognition from Raw Waveform with SincNet *2018 IEEE Spoken Language Technology Workshop (SLT)* pp 1021–8
[65] Clevert D-A, Unterthiner T and Hochreiter S 2015 Fast and accurate deep network learning by exponential linear units (elus) *arXiv preprint*
[66] Chollet F 2016 Xception: Deep Learning with Depthwise Separable Convolutions *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 1800–7
[67] Zhuang F, Qi Z, Duan K, Xi D, Zhu Y, Zhu H, Xiong H and He Q 2020 A Comprehensive Survey on Transfer Learning *arXiv:1911.02685 [cs, stat]*
[68] Kingma D P and Ba J 2017 Adam: A Method for Stochastic Optimization *arXiv:1412.6980 [cs]*
[69] Bergstra J, Bardenet R, Bengio Y and Kégl B 2011 Algorithms for hyper-parameter optimization *Advances in neural information processing systems* **24**
[70] Akiba T, Sano S, Yanase T, Ohta T and Koyama M 2019 Optuna: A Next-generation Hyperparameter Optimization Framework *arXiv:1907.10902 [cs, stat]*
[71] Frank Hutter, Holger Hoos, and Kevin Leyton-Brown 2014 An Efficient Approach for Assessing Hyperparameter Importance *Proceedings of the 31st International Conference on Machine Learning* ed Eric P. Xing and Tony Jebara (PMLR) pp 754–62
[72] Simonyan K, Vedaldi A and Zisserman A 2014 Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps *arXiv:1312.6034 [cs]*
[73] Campello R J G B, Moulavi D, Zimek A and Sander J 2015 Hierarchical Density Estimates for Data Clustering, Visualization, and Outlier Detection *ACM Trans. Knowl. Discov. Data* **10** 1–51
[74] McInnes L, Healy J and Astels S 2017 hdbscan: Hierarchical density based clustering *JOSS* **2** 205
[75] Benjamini Y and Hochberg Y 1995 Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing *Journal of the Royal Statistical Society. Series B (Methodological)* **57** 289–300
[76] Brancucci A, Lucci G, Mazzatenta A and Tommasi L 2009 Asymmetries of the human social brain in the visual, auditory and chemical modalities *Phil. Trans. R. Soc. B* **364** 895–914
[77] Hartikainen K M 2021 Emotion-Attention Interaction in the Right Hemisphere *Brain Sciences* **11** 1006
[78] Demaree H A, Everhart D E, Youngstrom E A and Harrison D W 2005 Brain Lateralization of Emotional Processing: Historical Roots and a Future Incorporating "Dominance" *Behavioral and Cognitive Neuroscience Reviews* **4** 3–20
[79] Simões M, Monteiro R, Andrade J, Mouga S, França F, Oliveira G, Carvalho P and Castelo-Branco M 2018 A Novel Biomarker of Compensatory Recruitment of Face Emotional Imagery Networks in Autism Spectrum Disorder *Front. Neurosci.* **12** 791
[80] Amaral C P, Simões M A and Castelo-Branco M S 2015 Neural Signals Evoked by Stimuli of Increasing Social Scene Complexity Are Detectable at the Single-Trial Level and Right Lateralized ed S Ben Hamed *PLoS ONE* **10** e0121970
[81] Hiremath C S, Sagar K J V, Yamini B K, Girimaji A S, Kumar R, Sravanti S L, Padmanabha H, Vykunta Raju K N, Kishore M T, Jacob P, Saini J, Bharath R D, Seshadri S P and Kumar M 2021 Emerging behavioral and neuroimaging biomarkers for early and accurate characterization of autism spectrum disorders: a systematic review *Transl Psychiatry* **11** 42
[82] Kamp-Becker I, Tauscher J, Wolff N, Küpper C, Poustka L, Roepke S, Roessner V, Heider D and Stroth S 2021 Is the Combination of ADOS and ADI-R Necessary to Classify ASD? Rethinking the "Gold Standard" in Diagnosing ASD *Front. Psychiatry* **12** 727308

[83] Picton T W 1992 The P300 Wave of the Human Event-Related Potential *Journal of Clinical Neurophysiology* **9** 456–79

CHAPTER 4: DESIGN OF A MULTI-SCALE CNN FOR P300 DECODING

The study reported in this chapter refers to the published journal paper entitled "A Lightweight Multi-Scale Convolutional Neural Network for P300 Decoding: Analysis of Training Strategies and Uncovering of Network Decision" D. Borra, S. Fantozzi and E. Magosso, *Frontiers in Human Neuroscience*, 2021. In particular, this chapter presents a novel structure for the deeper layers of a CNN for P300 decoding, by introducing a light and multi-scale temporal feature learning. Different variants (differing in the main hyper-parameters of the structure) of the novel CNN were investigated. Lastly, the decoding solutions were evaluated using multiple training strategies.

Convolutional neural networks (CNNs), which automatically learn features from raw data to approximate functions, are being increasingly applied to end-to-end analysis of electroencephalographic (EEG) signals, especially for decoding brain states in Brain-Computer Interfaces (BCIs). Nevertheless, CNNs introduce a large number of trainable parameters, may require long training times and lack in interpretability of learned features. The aim of this study is to propose a CNN design for P300 decoding with emphasis on its lightweight design while guaranteeing high performance, on the effects of different training strategies and on the use of post-hoc techniques to explain network decision. The proposed design, named MS-EEGNet, learns temporal features at two different time scales (i.e., multi-scale, MS) in an efficient and optimized (in terms of trainable parameters) way, and was validated on three P300 datasets. The CNN was trained using different strategies (within-participant and within-session, within-participant and cross-session, leave-one-subject-out, transfer learning) and was compared with several state-of-the-art (SOA) algorithms. Furthermore, variants of the baseline MS-EEGNet were analyzed, to evaluate the impact of different hyper-parameters on the performance. Lastly, saliency maps were used to derive representations of the relevant spatio-temporal features that drove CNN decisions. MS-EEGNet resulted the lightest CNN compared to the tested SOA CNNs, despite its multiple time scales, and significantly outperformed SOA algorithms. The post-hoc hyper-parameter analysis confirmed the benefits of the innovative aspects of our architecture. Furthermore, MS-EEGNet did benefit from transfer learning, especially using a low number of training examples, suggesting that our approach could be used in BCIs to accurately decode the P300 event while reducing calibration times. Representations derived from saliency maps matched the P300 spatio-temporal distribution, further validating the proposed decoding approach. The present study, by specifically addressing the aspects of lightweight design, transfer learning, interpretability, can contribute to advance the development of deep learning algorithms for P300-based BCIs.

4.1. INTRODUCTION

The P300 response is an attention-dependent event-related potential (ERP) first reported in electroencephalographic (EEG) signals by Sutton et al. [1]. This wave is characterized by a positive deflection peaking within the time window between 250 and 500 ms after the stimulus onset and it is mostly distributed on the scalp around the midline EEG electrodes (Fz, Cz, Pz), increasing its magnitude from the frontal to the parietal sites [2]. The P300 can be evoked in an oddball paradigm [3], where an infrequent deviant stimulus immersed in a sequence of frequent standard stimuli is presented to the user while he/she is attending to it (e.g., by counting how many times the rare event occurs). The rare events induce the P300 response; this response can be used as neural signal in EEG-based Brain-Computer Interfaces (BCIs), enabling a direct communication between the brain and the surroundings without the involvement of peripheral nerves or muscles [4]. One of the first P300-based BCIs was developed by Farwell and Donchin [3] using a visual stimulation in the oddball paradigm. These systems could be especially beneficial for patients suffering from motor neuron disease [5] to provide alternative ways of communication; furthermore, they may represent viable training tools for patients with attention deficits as recently reported in Amaral et al. [6] where a P300-based BCI paradigm was tested in patients suffering from Autism Spectrum Disorder (ASD) to improve their social attention.

Of course, a crucial aspect of a P300-based BCI is the decoding algorithm that translates the brain signals into classes (e.g., P300 and non-P300 classes). Machine learning (ML) techniques have been recognized to be powerful tools in learning discriminative patterns from brain signals. In recent years, deep learning, a branch of ML originally proposed in computer vision [7,8], has been applied to decoding problems of physiological signals such as electroencephalography, electromyography, electrocardiography and electrooculography [9]. At variance with more traditional ML approaches characterized by a separation between feature extraction, selection and classification stages [10], deep learning techniques automatically learn features from raw or light pre-processed inputs to maximize between-class discriminability, and finalize the decoding task in an end-to-end fashion.

Among deep learning techniques for classification, convolutional neural networks (CNNs) are widely used. These are specialized feed-forward neural networks involving the convolution operator to process data with a grid-like topology and are inspired by the hierarchical structure of the ventral stream of the visual system. Stacking neurons with local receptive field on top of others, creates receptive fields of individual neurons that increase in size in the deeper layers of the CNN and increases the complexity of the features the neurons respond to [11], realizing different levels of feature abstraction. This way, CNNs automatically learn hierarchical structured features from the input data, finalized to the classification. However, CNNs have some weaknesses: they introduce a large number of trainable parameters consequently requiring a large number of training examples, introduce many hyper-parameters (i.e., parameters that define the functional form of decoder), and the learned features are difficult to be interpreted.

The field of EEG classification (and in particular P300 classification) has been widely exploiting the advantages of CNNs [9,12]; at the same time, solutions to mitigate the

weaknesses of these algorithms have been proposed within this field, as reported in the state-of-the-art (SOA) description below.

In CNN-based EEG classification, EEG signals can be arranged into a 2D representation with electrodes along one dimension and time steps along the other, and fed as input to the CNN which predicts the corresponding label. CNN designs for EEG classification include both shallow and deep neural networks, and solutions have been proposed either performing spatial and temporal convolutions together (i.e., mixed spatio-temporal feature learning) or separately (i.e., unmixed spatio-temporal feature learning). Among the latter, several have been successfully applied to P300 classification [13–18] and generally have been proved to outperform traditional ML approaches. Cecotti and Graser [13] designed a CNN comprising 2 convolutional and 2 fully-connected layers to decode the P300 event. Remarkably, this was also the first attempt of CNN-based P300 decoding. Extensions of this architecture mainly focused on the increase of depth and inclusion of techniques such as batch normalization and dropout [14,15]. Moreover, Farahat et al. [16] proposed a dual-branched CNN (BranchedNet) that learns temporal features at two different time scales with parallel temporal convolutions, reporting an increase in performance with respect to a single-scale convolution. While these CNNs performed better than traditional ML techniques in P300 decoding, two aspects deserve attention: i) they learn spatial features (i.e., spatial convolution, performed across electrodes) and then temporal features in the next layers (i.e., temporal convolution, performed across time samples); ii) they do not address the challenge of reducing the number of trainable parameters. Regarding the first aspect, Shan et al. [17] pointed out that these architectures may lose useful raw temporal information related to the P300 event since temporal features are learned from spatially filtered signals instead from raw inputs. The authors proved that an architecture with the first layer performing a mixed spatio-temporal convolution (OCLNN) improved the decoding performance compared to the architecture proposed by Cecotti and Graser [13] and other variants [14,15]. Regarding the second aspect, recently Lawhern et al. [18] designed a shallow CNN for EEG decoding also applied to P300 detection (EEGNet). This design, besides performing temporal convolution in the first layer, uses separable and depthwise convolutions, i.e., convolutions specifically devoted to reduce the number of trainable parameters [19].

Remarkably, recently we proposed a CNN [20] based on the design of EEGNet, that won the P300 decoding challenge issued by the International Federation of Medical and Biological Engineering (IFMBE) in 2019, where the dataset (BCIAUT-P300) was a large multi-participant and multi-session collection of data. Our solution outperformed significantly a CNN derived from Manor and Geva [14] with a spatial convolutional layer as first layer, long short-term memories and traditional ML approaches [21]. These results further substantiate that CNNs including a temporal convolutional layer as first layer can represent advantageous solutions for P300 decoding, compared to traditional approaches and other CNN designs.

Techniques have been proposed for interpreting and understanding what the CNN has learned [22]; in the field of EEG classification, these are fundamental to validate a correct learning, checking that the learning system does not rely on artefactual sources but on neurophysiological features. These techniques explain the decoding decision taken by the CNN, i.e., features the CNN mainly relies on to discriminate among classes; this way, they represent tools to explore and analyze the underlying neurophysiology potentially characterizing new features (unknown so-far) and gaining insights into neural correlates of the

underlying phenomena. Montavon et al. [22] provided a definition for *explanation of CNN decision*: "the collection of features of the interpretable domain, that have contributed for a given example to produce a decision (e.g., classification or regression)". Among explanation techniques proposed in the computer vision domain [22], saliency maps [23] – simple representations reporting the gradient of a target class score with respect to each input pixel – were recently transposed to P300 decoding [16]. Furthermore, other techniques were adopted to understand CNNs for P300 decoding, such as temporal and spatial kernel visualizations [13,18], and kernel ablation tests [18]. In addition to these techniques, interpretable layers (where the learned features are directly interpretable without the needing of ad-hoc techniques) were recently applied to EEG decoding tasks [24-26].

Within this field of research, the aim of this study is to further contribute to the development of CNNs for EEG-based P300 decoding and to their analysis, with particular emphasis on the following aspects: keeping limited the number of trainable parameters (also referred to as model size) to realize lightweight CNNs suitable also for small datasets; assessing the effects of different learning strategies (including transfer learning) in view of the practical usage of these algorithms in BCIs; explaining the CNN decision i.e., the neurophysiological aspects that resulted in an optimal discriminability between classes. Specifically, the main contribution points are:

i. The realization of a CNN named MS-EEGNet (Multi-Scale EEGNet) combining two designs previously proposed in the literature with unique characteristics but treated separately, with the aim of jointly exploiting their respective strengths (see Section *MS-EEGNet*). On one hand, we adopted a branched architecture, in order to extract features at two different time scales, since this may improve the performance of P300 decoding (as suggested by Farahat et al. [16]). On the other hand, the branched solution would tend to increase the number of convolutional layers (since convolutions are replicated along each branch) and consequently the number of trainable parameters. Therefore, we adopted solutions to keep limited the number of trainable parameters, by limiting the overall number of convolutional layers (designing a shallow network) and at the same time implementing computationally efficient convolutions, such as depthwise and separable convolutions (as adopted by Lawhern et al. [18]). The latter are characterized by a reduced number of required multiplications, hence by a lower computational cost, and by a reduced number of trainable parameters compared to conventional convolutions (as those adopted by Farahat et al. [16]). In addition, learning compressed temporal representations in MS-EEGNet helped to further reduce the overall model size. This way, we proposed a multi-scale lightweight design. The so obtained network was then thoroughly analyzed to evaluate its performance and potentialities in view of practical applications (see points below).
ii. Analysis of the main hyper-parameters of the architecture, evaluating variant designs to investigate the role of multi-scale temporal feature learning (see Section *Alternative design choices of MS-EEGNet: changing hyper-parameters in the MST block*).
iii. Application of MS-EEGNet to 3 different datasets, to evaluate the proposed approach on variable sized datasets and on differently elicited P300 responses, comparing the performance with other SOA algorithms both CNNs and a traditional ML pipeline (see Sections 4.2.3 and 4.2.4).

iv. Training of MS-EEGNet with different strategies including transfer learning. The latter is of relevance as it could provide important benefits in practical BCI applications, alleviating the need of a large training set and reducing training times when using the CNN on a new user (see Section 4.2.5).
v. Application of an explanation technique based on saliency maps to derive the spatial and temporal features that drove MS-EEGNet decision (see Section 4.2.6).

4.2. MATERIALS AND METHODS

In this section, first we introduce the problem of EEG decoding via CNNs. Then, we describe the proposed architecture in its baseline and variant versions, P300 datasets, re-implemented SOA algorithms, training strategies and CNN explanation. Lastly, we illustrate the adopted statistical analyses.

CNNs were developed in PyTorch [27] and trained using a workstation equipped with an AMD Threadripper 1900X, NVIDIA TITAN V and 32 GB of RAM. Codes of MS-EEGNet are available at https://github.com/ddavidebb/P300_decoding_MS-EEGNet.

4.2.1. EEG decoding via CNNs

Let us consider an EEG dataset collected from many participants and recording sessions. Each single participant- and session-specific dataset is composed by many trials collected by epoching the continuous EEG recording respect to the onset of the stimulus (e.g., standard or deviant stimulus). Thus, each trial is associated to a specific class (e.g., non-P300 or P300 class), with a total of N_c classes. Indicating with $M^{(s,r)}$ the total number of trials for the s-th subject and the r-th recording session, the corresponding dataset can be formalized as $D^{(s,r)} = \left\{\left(X_0^{(s,r)}, y_0^{(s,r)}\right), \dots, \left(X_i^{(s,r)}, y_i^{(s,r)}\right), \dots, \left(X_{M^{(s,r)}-1}^{(s,r)}, y_{M^{(s,r)}-1}^{(s,r)}\right)\right\}$. $X_i^{(s,r)} \in \mathbb{R}^{C \times T}$ represents the pre-processed EEG signals of the i-th trial ($0 \leq i \leq M^{(s,r)} - 1$), indicating with C the number of electrodes and T the number of time steps. $y_i^{(s,r)}$ is the label associated to $X_i^{(s,r)}$, i.e., $y_i^{(s,r)} \in L = \{l_0, \dots, l_{N_c-1}\}$. In the particular case of P300 decoding, i.e., discrimination between standard and deviant trials, $N_c = 2$ and $L = \{l_0, l_1\} = \{\text{"non-P300"}, \text{"P300"}\}$.

The objective decoding problem can be formalized as the optimization of a parametrized classifier f implemented by a CNN, $f\left(X_i^{(s,r)}; \theta\right): \mathbb{R}^{C \times T} \rightarrow L$ with parameters θ, learning from a training set to assign the correct label to unseen EEG trials. Therefore, in the following, we refer to $X_i^{(s,r)}$ as the CNN input, represented as a 2D matrix of shape (C, T) with time steps along the width and electrodes along the height. Lastly, each dataset $D^{(s,r)}$ was divided into a training set, used to optimize the parameters contained in the array θ, and a test set, used to evaluate the algorithm on unseen data. Furthermore, a separate validation set need to be extracted from the training set to define a stop criterion of the optimization. As described in Section 4.2.3, here we used 3 datasets: dataset 1 was a large public dataset where each participant performed different recording sessions, while datasets 2 and 3 were two small private datasets, where each participant performed one single recording session.

4.2.2. The proposed CNN and its variants

MS-EEGNet

The proposed shallow architecture was composed by 3 fundamental blocks, each consisting of many layers. A schematic representation of the CNN is reported in Figure 4.1. The spatio-temporal (ST) block extracted temporal and spatial features from the input EEG signals via temporal and spatial convolutional layers, respectively. Downstream, the multi-scale temporal (MST) block used lightweight parallel temporal convolutions to extract temporal patterns at

different scales from the feature maps provided by the previous block. Lastly, the multi-scale activations were provided to the fully-connected (FC) block that finalized the decoding task using a single fully-connected layer.

In all layers except the last two, the output was a collection of spatio-temporal feature maps and its shape can be described by a tuple of 3 integers, with the first integer indicating the number of feature maps, the second and third integers representing the number of spatial and temporal samples within each map, respectively. In the following, to describe the CNN we will refer to the hyper-parameters of the involved layers. Each convolutional layer is characterized by the number of convolutional kernels (K), kernel size (F), stride size (S) and padding size (P). In addition, depthwise convolution introduced also a depth multiplier (D) specifying the number of kernels to learn for each input feature map. Hyper-parameters will be denoted by a superscript and a subscript. The superscript indicates the specific block the layer belongs to, using acronyms "ST", "MST_0", "MST_1", "FC", where the index in the multi-scale temporal block (MST) discriminates between the two scales (in general MST_i, where $0 \leq i \leq N_b - 1$ and N_b denotes the number of parallel branches). The subscript indicates which convolutional layer inside the block the hyper-parameters refer to (convolutional layers inside each block were labeled with an increasing index, starting from 0). Lastly, pooling layers were described by pool size (F_p) and pool stride (S_p), with the corresponding superscript. Both convolutions and poolings were 2D; therefore, F, S, P, F_p, S_p were tuples of two integers: the first referred to the spatial dimension, while the second to the temporal dimension. Lastly, the number of time samples changed across pooling operators and was denoted with T_p. Regarding the single fully-connected layer included in the classification block, the number of neurons was denoted with N^{FC} and represented the number of classes to decode (N_c).

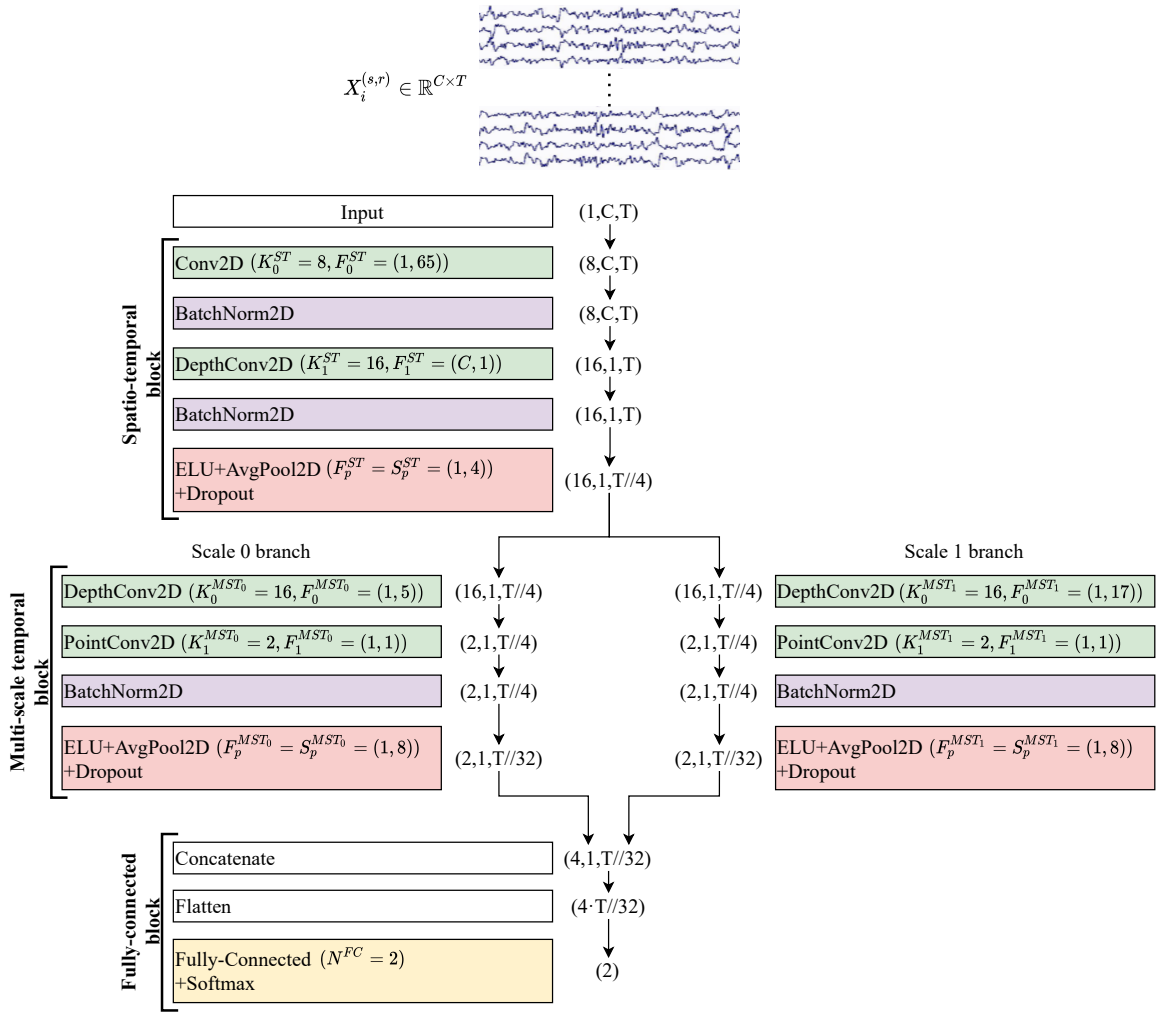


Figure 4.1 – Structure of MS-EEGNet. Layers are represented by coloured rectangles, reporting the layer name and main hyper-parameters. The tuple outside each rectangle represents the output shape of each layer. For all outputs except the last two (Flatten and Fully-connected+Softmax), tuples are composed by three numbers representing the number of the feature maps (channel dimension), the number of spatial samples and the number of temporal samples within each map, respectively. The input layer provides an output of shape $(1, C, T)$ as it just replicates the original input matrix with shape (C, T), providing a single feature map as output. The temporal dimension changed from T to $T//32$ along the entire CNN (where the symbol // indicates the floor division operator) due to the average pooling operations. See Sections 4.2.1 and 4.2.2 for the meaning of symbols and see Table 4.1 for further details.

MS-EEGNet was analyzed in a baseline version and in many variants by adopting a post-hoc hyper-parameter evaluation procedure on the main MST block hyper-parameters. The baseline version is described in the current section, where the structure and function of each block are presented, while the variants in Section *Alternative design choices of MS-EEGNet: changing hyper-parameters in the MST block*.

i. Spatio-temporal block. This was designed to learn temporal and spatial features separately. At first, a temporal convolutional layer was included, learning ${K_0}^{ST} = 8$ temporal kernels with filter size ${F_0}^{ST} = (1{,}65)$, unitary stride and zero-padding ${P_0}^{ST} = (1{,}32)$ to preserve the number of input temporal samples. Then, ${D_1}^{ST} = 2$ spatial filters of size $(C, 1)$ were learned for each temporal feature map in a spatial depthwise convolutional layer, with unitary stride and without zero-padding [18,20]. Thus, a total

number of ${K_1}^{ST} = {K_0}^{ST} \cdot {D_1}^{ST} = 16$ spatial filters were learned and constrained to have a norm upper bounded by $c = 1$ (kernel max-norm constraint) as in previous studies [18,20,28]. The feature maps of this layer were not fully-connected with the feature maps of the previous layer. This not only reduced the number of trainable parameters, but also allowed a more straightforward spatio-temporal feature learning. Indeed, each group of D_1 spatial filters was related to a specific temporal filter [18] (i.e., to specific spectral information). Furthermore, the output activations of the temporal and spatial convolutional layers were normalized via batch normalization [29]. Downstream the spatial depthwise convolution and its associated batch normalization, the neurons were activated via an Exponential Linear Unit (ELU) non-linearity [30], i.e., $f(x) = x, x > 0$ and $f(x) = \alpha \cdot (exp(x) - 1), x \leq 0$. We adopted this activation function since it was proved to allow faster and more noise-robust learning than other non-linearities [30] and to outperform other activation functions when using CNNs with EEG signals [31]. The α hyper-parameter controls the saturation value for negative inputs and $\alpha = 1$ was set here. Then, an average pooling layer was introduced to reduce the size of the activations along the temporal dimension from T to ${T_p}^{ST}$, with a pool size of ${F_p}^{ST} = (1,4)$ and pool stride of ${S_p}^{ST} = (1,4)$, providing activations sampled at 1/4 of the sampling frequency of the signals (32 Hz when using signals extracted from dataset 1 and approx. 31.3 Hz from datasets 2 and 3). Lastly, a dropout layer [32] (with a different dropout rate p depending on the training strategy adopted, see Section 4.2.5) was added.

ii. Multi-scale temporal block. This block was designed to learn how to summarize along the temporal dimension the feature maps provided by the spatio-temporal block. Differently from EEGNet where features at a single time scale were learned at this stage, here the features were learned at N_b different time scales, inspired from the design of the Inception modules [33]. In the baseline MS-EEGNet $N_b = 2$, thus, two different sets of short and large kernels were separately learned in the two parallel branches, respectively. This was accomplished via 2 parallel temporal depthwise convolutional layers with a unitary depth multiplier, i.e., ${D_0}^{MST_0} = {D_0}^{MST_1} = {D_0}^{MST} = 1$ and ${K_0}^{MST_0} = {K_0}^{MST_1} = {K_0}^{MST} = {K_1}^{ST} \cdot {D_0}^{MST}$, and with different kernel sizes in the two branches extracting a summary of roughly 150 ms (${F_0}^{MST_0} = (1,5)$) and 500 ms (${F_0}^{MST_1} = (1,17)$), for each input feature map. That is, each output feature map was a sort of weighted moving average of the input feature map using moving windows of 2 different lengths, approximately 150 ms and 500 ms, respectively (referred to as scales). The large kernel size was chosen to match the temporal kernel size used in the single-scale branch of EEGNet [20]. The small kernel size was chosen so that the ratio between the small and large kernel was approximately the same as in BranchedNet ($r^{MST} = {}^{1}/_{4}$), keeping odd kernel size (i.e., 500 ms/4=125 ms=4 samples at 32 Hz, approximated to 5 samples to have an odd integer). The small and large temporal filters should be able to learn high and low frequency patterns from the input, respectively [34]. Here, unitary stride and zero-padding of ${P_0}^{MST_0} = (0,2)$ and ${P_0}^{MST_1} = (0,8)$ were adopted, preserving the number of the input temporal samples. After each depthwise convolutional layer, a pointwise convolutional layer (${F_1}^{MST_0} = {F_1}^{MST_1} = {F_1}^{MST} = (1,1)$) was added to learn

how to optimally combine the feature maps at a specific time scale, with unitary stride and without zero-padding. At variance with BranchedNet [16] where convolutions were not designed to keep limited the number of trainable parameters, the proposed multi-scale temporal block was designed using separable convolutions (i.e., depthwise convolution followed by pointwise convolution), with the specific aim of reducing the training parameters. In this same perspective, the number of output feature maps was set as low as ${K_1}^{MST_0} = {K_1}^{MST_1} = {K_1}^{MST} = 2$ in each branch, learning a compressed representation of the input feature maps (i.e., the 16 input feature maps provided by the depthwise convolutional layer were recombined into only 2 different feature maps, for each branch). Then, for each branch, the output activations of the pointwise convolutional layer were normalized via batch normalization [29] and activated with an ELU non-linearity ($\alpha = 1$). Finally, an average pooling layer was introduced with a pool size of ${F_p}^{MST_0} = {F_p}^{MST_1} = {F_p}^{MST} = (1, 8)$ and pool stride of ${S_p}^{MST_0} = {S_p}^{MST_1} = {S_p}^{MST} = (1, 8)$ to reduce the temporal dimension from ${T_p}^{ST}$ to ${T_p}^{MST}$, followed by a dropout layer [32] (with a different dropout rate p depending on the training strategy adopted, see Section 4.2.5).

iii. Fully-connected block. This block was devoted to produce the output probabilities from the feature maps provided by the multi-scale temporal block. The input feature maps were concatenated together along the feature map dimension and unrolled along one single dimension via a flatten layer. Then, this multi-scale feature vector was given as input to a fully-connected layer with $N^{FC} = N_c = 2$ neurons (associated to the P300 and non-P300 classes). These 2 outputs were transformed via a softmax activation function to obtain the conditional probabilities $p\left(l_k \middle| X_i^{(s)}\right)$, $k = 0, 1$.

A more detailed description of the structural hyper-parameters and of the number of trainable parameters of the baseline version of MS-EEGNet can be found in Table 4.1. The overall number of trainable parameters (or model size) and the training time (or computational time) of the baseline MS-EEGNet are reported in Table 4.2. Note that in this table, these variables are reported also for the variant designs of MS-EEGNet (see Section *Alternative design choices of MS-EEGNet: changing hyper-parameters in the MST block*) and for the examined SOA CNNs (see Section 4.2.4).

Table 4.1 – Architecture details of MS-EEGNet. Each layer is provided with its name, main hyper-parameters and number of trainable parameters. See Sections 4.2.1 and 4.2.2 for the meaning of symbols. The total number of trainable parameters was 1154, when using signals from the dataset 1, and 1210, when using signals from datasets 2 and 3. In all layers, where not specified, stride (S) and padding (P) were set to $(1,1)$ and $(0,0)$, respectively.

Block	Layer name	Hyper-parameters	Number of trainable parameters
	Input	$K_0 = 1$	0
ST	Conv2D	${K_0}^{ST} = 8$, ${F_0}^{ST} = (1,65)$, ${P_0}^{ST} = (0,32)$	${F_0}^{ST}[0] \cdot {F_0}^{ST}[1] \cdot {K_0}^{ST} \cdot K_0$
	BatchNorm2D	$m = 0.99$	$2 \cdot {K_0}^{ST}$
	Depthwise-Conv2D	${D_1}^{ST} = 2$, ${K_1}^{ST} = {K_0}^{ST} \cdot {D_1}^{ST}$, ${F_1}^{ST} = (C, 1)$, kernel max norm=1	${F_1}^{ST}[0] \cdot {F_1}^{ST}[1] \cdot {K_0}^{ST} \cdot {D_1}^{ST}$
	BatchNorm2D	$m = 0.99$	$2 \cdot {K_1}^{ST}$
	ELU	$\alpha = 1$	0
	AvgPool2D	${F_p}^{ST} = {S_p}^{ST} = (1,4)$	0
	Dropout	$p = 0.25$ or $p = 0.5$	0
MST scale 0	Depthwise-Conv2D	${D_0}^{MST_0} = 1$, ${K_0}^{MST_0} = {K_1}^{ST} \cdot {D_0}^{MST_0}$, ${F_0}^{MST_0} = (1,5)$, ${P_0}^{MST_0} = (0,2)$	${F_0}^{MST_0}[0] \cdot {F_0}^{MST_0}[1] \cdot {K_1}^{ST} \cdot {D_0}^{MST_0}$
	Pointwise-Conv2D	${K_1}^{MST_0} = 2$, ${F_1}^{MST_0} = (1,1)$	${F_1}^{MST_0}[0] \cdot {F_1}^{MST_0}[1] \cdot {K_1}^{MST_0} \cdot {K_0}^{MST_0}$
	BatchNorm2D	$m = 0.99$	$2 \cdot {K_1}^{MST_0}$
	ELU	$\alpha = 1$	0
	AvgPool2D	${F_p}^{MST_0} = {S_p}^{MST_0} = (1,8)$	0
	Dropout	$p = 0.25$ or $p = 0.5$	0
MST scale 1	Depthwise-Conv2D	${D_0}^{MST_1} = 1$, ${K_0}^{MST_1} = {K_1}^{ST} \cdot {D_0}^{MST_1}$, ${F_0}^{MST_1} = (1,17)$, ${P_0}^{MST_1} = (0,8)$	${F_0}^{MST_1}[0] \cdot {F_0}^{MST_1}[1] \cdot {K_1}^{ST} \cdot {D_0}^{MST_1}$
	Pointwise-Conv2D	${K_1}^{MST_1} = 2$, ${F_1}^{MST_1} = (1,1)$	${F_1}^{MST_1}[0] \cdot {F_1}^{MST_1}[1] \cdot {K_1}^{MST_1} \cdot {K_0}^{MST_1}$
	BatchNorm2D	$m = 0.99$	$2 \cdot {K_1}^{MST_1}$
	ELU	$\alpha = 1$	0
	AvgPool2D	${F_p}^{MST_1} = {S_p}^{MST_1} = (1,8)$	0
	Dropout	$p = 0.25$ or $p = 0.5$	0
FC	Concatenate		0
	Flatten		
	Fully-Connected	$N^{FC} = 2$	$N^{FC} \cdot ({T_p}^{MST_0} \cdot {K_1}^{MST_0} + {T_p}^{MST_1} \cdot {K_1}^{MST_1} + 1)$
	Softmax		0

Alternative design choices of MS-EEGNet: changing hyper-parameters in the MST block

In addition to the baseline MS-EEGNet described previously, we evaluated other alternative designs to better investigate the behavior of the proposed MST block, by modifying some hyper-parameters (HPs), once at a time. In the following, the alternative designs are described and indicated via the modified HP: $HP_{variant}$ vs. $HP_{baseline}$.

i. $N_b = \{1,3\}$ vs. $N_b = 2$: use of one or three branches. In this post-hoc analysis, we studied whether the proposed dual-scale temporal feature learning was beneficial compared to the traditional single-scale learning ($N_b = 1$) and which scale was able to learn more relevant class-discriminative temporal features. To this aim, MS-EEGNet was modified either by removing the short-scale (scale 0) leaving only the large-scale branch ($N_b =$

1(large)) or the large-scale (scale 1) leaving only the short-scale branch ($N_b = 1(\text{short})$). It is worth noticing that the single-scale variant design $N_b = 1$ (large) did not correspond to the EEGNet adaptation used in [20] since here we adopted compressed representations in separable convolutional layers. In addition, we studied whether a third time scale ($N_b = 3$) could be useful, by modifying MS-EEGNet with the inclusion of an additional time scale between the ones of the baseline version: this variant learned summaries of about 125, 250, 500 ms, corresponding to kernel sizes in the MST block of $\mathrm{F_0}^{\mathrm{MST_0}} = (1,5)$, $\mathrm{F_0}^{\mathrm{MST_1}} = (1,9)$, $\mathrm{F_0}^{\mathrm{MST_2}} = (1,17)$, respectively.

ii. ${F_0}^{MST_0} = (1,9)$ vs. ${F_0}^{MST_0} = (1,5)$: enlarging the kernel size in the short-scale branch (scale 0 in Table 4.1). This was performed to evaluate the effect of a different ratio between the short- and large-scale of the MST block than the one adopted in the baseline MS-EEGNet. Specifically, $r^{MST} = {}^1/_2$ vs. $r^{MST} = {}^1/_4$ leading to 500 ms/2=250 ms=8 samples at 32 Hz, approximated to 9 samples to have odd integer.

iii. ${K_1}^{MST} = \{1,8,16\}$ vs. ${K_1}^{MST} = 2$: different number of feature maps in the pointwise convolutions. In particular, ${K_1}^{MST}$ was set to 1 in each branch, in order to analyze whether the learning of a single recombination of the input feature maps was enough to provide accurate decoding performance. In addition, ${K_1}^{MST}$ was set to 8 in each branch, in order to analyze another compressed representation, but maintaining the total number of feature maps across the two different time scales unchanged as in the MST input (i.e., 8 feature maps in each branch, resulting into 16 feature maps across the 2 scales, as in the input of the MST block). Lastly, ${K_1}^{MST}$ was set to 16 in each branch, corresponding to a condition where no compressed representation was learned in either branch.

iv. *deepMST* vs. *MST:* increasing the depth of MST block. This was performed to evaluate the effect on the performance of an increased depth in the MST block (and thus, learning more non-linear dependencies), while maintaining the same overall receptive field of the neurons in the temporal domain. In each branch we added another depthwise convolutional layer after the first one; however, in order to maintain the same receptive field as when using a single depthwise convolutional layer in the baseline MST block, the kernel size of each depthwise convolutional layer was halved with respect to the baseline values, i.e., ${F_0}^{MST_0} = {F_1}^{MST_0} = (1,3)$ and ${F_0}^{MST_1} = {F_1}^{MST_1} = (1,9)$. After the second depthwise convolutional layer, the pointwise convolutional layer was added (${F_2}^{MST_0} = {F_2}^{MST_1} = {F_2}^{MST} = (1,1)$), and the rest of the block was maintained unchanged as in the baseline version.

Overall, 8 variants were designed by changing a specific hyper-parameter value of MS-EEGNet while keeping all the other hyper-parameters as in the baseline MS-EEGNet. These alternative designs were trained with a within-participant and within-session strategy (as it is the most common strategy adopted in literature) and compared with MS-EEGNet trained with the same strategy. Lastly, the number of trainable parameters and training time are reported in Table 4.2 for each variant design.

Table 4.2 – Number of trainable parameters, also denoted as model size in the text, and training time (referred to the WS strategy), also denoted as computational time, of the baseline MS-EEGNet, MS-EEGNet variants and SOA CNNs when using signals from the dataset 1 and datasets 2-3. These values were reported for deep learning-

based decoders to provide a more complete comparison between the proposed CNN and the SOA. For each CNN, among dataset 1 and datasets 2-3, the different number of parameters resulted from the different number of EEG channels ($C = 8$ for dataset 1 and $C = 12$ for datasets 2-3, see Section 4.2.3) and time samples considered ($T = 140$ for dataset 1 and $T = 113$ for datasets 2-3, see Section 4.2.3), while the different training time resulted from the different number of training examples (1280 trials and 240 trials, for each participant and each session respectively for dataset 1 and datasets 2-3, see Section 4.2.3). In addition, the percentage difference (Δ) of trainable parameters and training time between SOA CNNs or MS-EEGNet variants ("other" condition) and the baseline MS-EEGNet ("baseline" condition) is reported, i.e., $100 \cdot (value_{other} - value_{baseline})/value_{baseline}$.

Algorithm	**Trainable parameters (dataset 1/datasets 2-3)**		**Training time (dataset 1/datasets 2-3)**	
	Value	**Δ (%)**	**Value (ms/epoch)**	**Δ (%)**
Baseline MS-EEGNet	1154/1210	-	220/45.5	-
MS-EEGNet variants				
$N_b = 1(large)$	1022/1082	-11.4/-10.6	195/38.1	-11.4/-16.3
$N_b = 1(short)$	830/890	-28.1/-26.5	172/38.4	-21.8/-15.6
$N_b = 3$	1350/1402	17.0/15.9	282/50.8	28.2/11.6
${F_0}^{MST_0} = (1,9)$	1218/1274	5.5/5.3	221/46.3	0.5/1.8
${K_1}^{MST} = 1$	1102/1162	-4.5/-4.0	224/45.0	1.8/-1.1
${K_1}^{MST} = 8$	1466/1498	27.0/23.8	287/46.1	30.5/1.3
${K_1}^{MST} = 16$	1882/1882	63.1/55.5	240/45.0	9.0/-1.1
deepMST	1202/1258	4.2/4.0	295/47.0	34.1/3.3
SOA CNNs				
EEGNet	1386/1418	20.1/17.2	186/40.5	-15.5/-11.0
BranchedNet	5418/7954	369/557	250/50.3	13.6/10.5
OCLNN	1650/1874	43.0/54.9	96.2/22.9	-56.3/-49.7

4.2.3. Data and pre-processing

Dataset 1

The first dataset is BCIAUT-P300, a public benchmark released for the IFMBE 2019 scientific challenge (available at https://www.kaggle.com/disbeat/bciaut-p300) [21] consisting of a larger number of examples than other public benchmark [35,36] or private [16,18,37] datasets. Signals were recorded from 15 participants (all males, age of 22±5 years, mean ± standard deviation) with ASD during 7 recording sessions (for a total of 4 months) while testing a P300-based BCI [6]. The paradigm consisted in the participant paying attention to one among 8 objects randomly flashed in a virtual scene, with the P300 stimuli corresponding to the flashing of the attended object (this was repeated several times for each different attended object). For each participant and recording session, 1600 trials were recorded during the calibration stage (training set) and 2838 trials were recorded during the online stage (test set), on average.

Signals were recorded at 250 Hz from 8 electrodes: C3, Cz, C4, CPz, P3, Pz, P4, POz. The reference was placed at the right ear and the ground at AFz. These signals were acquired notch filtered at 50 Hz and then pass-band filtered between 2 and 30 Hz [21]. EEG signals were pre-processed as in previous studies [20,38]. In particular, epochs were selected from -100 to 1000 ms relative to the event stimulus and signals were downsampled to 128 Hz to reduce the number of time steps to be processed in the CNN. Architectures were trained as described in Section 4.2.5 using the training set of the competition for each session, while the test set was

used to test the algorithms. From each participant- and session-specific training set, a validation set of 20% of the total training set was extracted (corresponding to 320 trials) to perform early stopping, while the remaining percentage of the total training set (corresponding to 1280 trials) was used to optimize the architectures.

Datasets 2 and 3

The second dataset was collected from 7 participants (all males, age 25±8 years) recorded in an auditory oddball study during a single recording session and the third dataset was collected from 7 participants (5 males, age 22±0.4 years, different participants than dataset 2) recorded in a visual oddball study during a single recording session. All participants were healthy volunteers not reporting psychological or hearing disorders. Both experiments were approved by the Bioethics Committee of the University of Bologna (file number 29146, year 2019) and were conducted in a controlled laboratory environment.

The auditory oddball paradigm consisted of 400 tones presented to the participant through a speaker, with the standard and deviant stimuli differing by the frequency of the tones (500 Hz and 1000 Hz respectively). The visual oddball paradigm consisted of 400 stimuli presented to the participant through a bicolour LED with the standard and deviant stimuli differing by the LED colour (blu and red, respectively). In both paradigms, each stimulus was reproduced for 56 ms followed by a pause of 944 ms (inter-stimuli interval); thus, each trial lasted 1 s. This paradigm was similar to the one adopted in [39]. Furthermore, in each paradigm, a total number of 325 standard and 75 deviant stimuli were presented to the participant in a randomized order. Thus, for participant, a total number of 400 trials were available, with a class imbalance ratio of 75:325 for the P300 and non-P300 classes. While listening to the tones or while looking to the LED, participants were seated on a comfortable chair in front of a button with their eyes opened and were instructed to respond to the deviant stimuli by pressing the button with their right index as quickly as possible, minimizing other movements.

Signals of both datasets 2 and 3 were recorded at 125 Hz using a portable EEG recording system (OpenBCI system, using Cyton and Daisy Biosensing boards) from 12 electrodes: C3, Cz, C4, CP5, CP1, CPz, CP2, CP6, P3, Pz, P4, PO3, PO4. The reference was placed at the right earlobe and the ground at the left earlobe. The same pre-processing was adopted for datasets 2 and 3. In particular, signals were band-pass filtered between 2 and 30 Hz with a zero-phase 2nd order filter and epochs were extracted from -100 to 800 ms relative to the stimulus onset. For datasets 2 and 3, the architectures were trained as described in Section 4.2.5, using a 4-fold cross-validation scheme. Therefore, in each fold, each participant-specific dataset was divided into a training (75%) and test (25%) set, corresponding to 300 and 100 trials, respectively. Lastly, a validation set of 20% of the training set (corresponding to 60 trials) was extracted to perform early stopping, while the remaining percentage (corresponding to 240 trials) was used to optimize the architectures.

As described in Section 4.2.1, $X_i^{(s,r)} \in \mathbb{R}^{C \times T}$ represented the CNN input. From the previous dataset descriptions, $C = 8$ for dataset 1 and $C = 12$ for datasets 2 and 3, while $T = 140$ for dataset 1 and $T = 113$ for datasets 2 and 3.

4.2.4. State-of-the-art algorithms

The proposed baseline architecture was compared to other SOA algorithms, including the winning algorithm of the IFMBE 2019 challenge based on EEGNet [20], BranchedNet [16] and OCLNN [17]. The first was a single-branched CNN performing the temporal convolution in the first layer. The second one was a dual-branched CNN exploiting parallel temporal convolutions but, at variance with the architecture proposed here, performed the spatial convolution in the first layer and did not use optimized convolutions aimed to keep limited the number of trainable parameters, resulting in a less parsimonious multi-scale CNN. OCLNN was a CNN performing a mixed spatio-temporal convolution in the first layer without using optimized convolutions. To allow a more complete comparison between MS-EEGNet and other deep learning-based decoders, the number of trainable parameters and training time of SOA CNNs are summarized in Table 4.2.

In addition to these SOA CNNs, we re-implemented xDAWN+RG, a ML pipeline for P300 decoding. In particular, this solution included a combination of xDAWN spatial filtering [40,41], Riemannian Geometry [42], L_1 feature regularization and classification based on an Elastic Net regression.

Details about SOA CNNs and xDAWN+RG can be found in Supplementary Materials (Sections 4.6.1 and 4.6.2).

4.2.5. Training

MS-EEGNet was trained adopting different training strategies.

i. Within-participant and within-session training (WS). For each participant and each session, EEG signals (see Section 4.2.3) were used to train, validate and test a participant-specific and session-specific CNN. In addition, we also trained CNNs using only a fraction of the participant- and session-specific training set, simulating practical cases of reduced numbers of available calibration trials, and investigated how the performance changed; this is an important issue in the perspective of limiting the calibration time in practical applications. Reduced training sets were defined by extracting 15, 30, 45, 60% of the total training set in the corresponding session (corresponding to 192, 384, 576, 768 training trials for dataset 1, and 48, 96, 144, 192 trials for datasets 2 and 3) maintaining the class imbalance characterizing each dataset. For each architecture, 105 (15 participants *7 sessions per participant) CNNs were trained for dataset 1, while 7 (7 participants * 1 session per participant) CNNs were trained for datasets 2 and 3. The WS strategy (with 100% of training trials) was adopted also with SOA algorithms and to perform the post-hoc hyper-parameter evaluation.
ii. Within-participant and cross-session training (CS). This training strategy was adopted only for the dataset 1 due to its multi-session dimension and was the strategy used in our winning solution in the IFMBE 2019 challenge [20] using the same dataset. For each participant, an overall training set, and an overall validation set were obtained by considering all the session-specific training and validation sets belonging to that particular participant. Then, these overall sets were used to train and validate a participant-specific CNN incorporating inter-session variability. It is worth noticing that this participant-specific CNN was then tested separately over each session-specific test

set (relative to that participant), for consistency with the test procedure adopted in i). For each architecture, 15 CNNs were trained for the dataset 1. This strategy was adopted also with SOA algorithms.

iii. Leave-one-subject-out training (LOSO). The EEG signals of one participant (i-th participant) were held back, and the training and validation sets were obtained by collecting EEG signals from all the session-specific training and validation sets of the remaining participants (j-th participants $\forall j, j \neq i$). Thus, for each held back participant ($\forall i$) an architecture was trained and validated with signals extracted from 14 participants for dataset 1 and from 6 participants for datasets 2 and 3. The so obtained network was then tested separately over each session-specific test set of the held back participant, consistently with the testing procedure in i) and ii). The residual signals of the held back participant not used in the testing procedure remained unused (i.e., 0% of the held back participant's dataset was used to train and validate the model); that is, LOSO models did not learn from examples of the held back participant. This training strategy led to a CNN incorporating inter-participant and (in case of dataset 1) inter-session variabilities. For each architecture, 15 CNNs were trained for dataset 1, while 7 CNNs were trained for datasets 2 and 3. This strategy was adopted also with SOA algorithms. Lastly, to design LOSO models incorporating the knowledge from a variable number of participants, we additionally performed trainings extracting signals from a random subset of participants, i.e., using 10, 6, 2 participants for dataset 1, and using 4, 2 participants for datasets 2 and 3. Thus, the performed LOSO strategy was named "LOSO-M", where M is the number of participants used ($M = \{14,10,6,2\}$ when using signals from dataset 1, while $M = \{6,4,2\}$ for datasets 2 and 3). It is worth noticing that the LOSO-14 strategy for dataset 1 and LOSO-6 strategy for datasets 2 and 3 corresponded to the conventional LOSO strategies for these datasets.

iv. Transfer learning (TL) on single sessions (WS). As in WS strategy (point i), for each participant and each session, EEG signals (see Section 4.2.3) were used to train, validate and test a participant- and session-specific CNN. Differently from the WS strategy where the trainable parameters were initialized randomly, in the TL-WS strategy the parameters were initialized from the ones obtained with LOSO trainings when the specific participant of interest was held back. Therefore, the knowledge learned in the LOSO strategy (using training examples sampled from many participants except the held back participant) was transferred on the held back participant. Then, a fraction of the session-specific training set of the held back participant was used as training set, using the same percentages as in WS strategy (point i); in this way we compared the performance of WS and TL-WS strategy, to investigate if and to what extent TL-WS strategy outperformed WS strategy with a reduced number of calibration trials. For each architecture, 105 CNNs were trained for dataset 1, while 7 CNNs were trained for datasets 2 and 3.

The transfer learning strategy reflects a practical situation in which a new user approaches the BCI system in a new session, and a calibration phase – as short as possible – is needed to obtain an accurate participant-specific decoder. Therefore, a pre-trained model that incorporates both inter-participant and inter-session variabilities as obtained with the LOSO strategy, could be a better initialization point respect to the random one (as used in the WS training strategy), leading to a performance improvement especially

when using only a small number of training examples of a new user in a new recording session.

The adopted training strategies had a different definition of the training set; however, in all cases, CNNs were tested on the same participant-specific and session-specific test sets, allowing a fair comparison across different training strategies. In this study the adopted metric to quantify the performance for the P300 decoding task at the trial-level was the Area Under the ROC Curve (AUC), as done previously [18], and was computed on each participant- and session-specific test set.

EEG signals of the training, validation and test sets were standardized computing the mean and variance on the training set. Regarding the TL-WS strategy, the first and second moments were computed on the training set used to train the pre-trained models. Except for the TL-WS strategy, in which the trainable parameters were initialized from the pre-trained models, in the other training approaches the weights were randomly initialized adopting a Xavier uniform initialization scheme [43] and biases were initialized to zero.

The optimization was performed by minimizing the negative log likelihood or, equivalently, the cross-entropy between the empirical probability distribution defined by the training labels and the probability distribution defined by the model. Adaptive moment estimation (Adam) [44] was used as optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ for computing the running averages of the gradient and its square, and $\varepsilon = 10^{-8}$ to improve numerical stability. The learning rate was set to $lr = 10^{-3}$ for WS, CS and LOSO strategies, while for the TL-WS strategy the optimizer state was set to the one of the pre-trained models. To address the class imbalance, a single mini-batch of data was composed by a proportion of 50–50% of the two classes, randomly selecting the trials within the dataset as done in [20]. The mini-batch size and the maximum number of epochs were set to 64 and 500, respectively, and early stopping was performed by interrupting the optimization when the validation loss did not decrease for 50 consecutive epochs.

In addition to early stopping which acts as a regularizer, other regularizer mechanisms were integrated into MS-EEGNet as mentioned in Section *MS-EEGNet*, comprising batch normalization [29], with a momentum term of $m = 0.99$ and $\varepsilon = 1e - 3$ for numerical stability, dropout [32], with a dropout probability of 0.5 for WS and TL-WS trainings and 0.25 for CS and LOSO trainings, and kernel max-norm constraint.

4.2.6. Explaining P300 decision: gradient-based representations

MS-EEGNet decision was explained using saliency maps, post-hoc (i.e., obtained once the CNN training has ended) gradient-based representations proposed by Simonyan et al. [23] to quantify the importance of neurons belonging to a target layer of interest (commonly the input layer) for a specific class. These representations are commonly used to explain CNN decision when decoding EEG [16,26,28] and offer the advantage of requiring the sole computation of the backpropagation. Of course, other more advanced techniques, such as layer-wise relevance propagation (LRP), can represent a valid alternative but they introduce many factors that affect the representations, such as the propagation rule (e.g., αβ rule) and propagation parameters (e.g., α and β) [22], whose setting would require preliminary deep investigations. Hence, we preferred to adopt saliency maps. Here, these were computed by backpropagating the gradient of the P300 class score (i.e., the output related to the P300 neuron, immediately before the

softmax activation) back to the input layer (i.e., the neurons corresponding to the input spatio-temporal samples), when P300 trials belonging to the test set were fed as input to the CNN. Thus, each resulting saliency map was a spatio-temporal representation associated to a test trial, quantifying how much each spatio-temporal input sample affects the P300 class score, i.e., how much the P300 class score changes with respect to a small change in the input EEG signals. For each dataset, these representations were computed using MS-EEGNet trained with the LOSO strategy, as this strategy was more likely to enhance input samples relevant for the decoding task compared to WS/CS trainings [16]. Indeed, during LOSO trainings, models were fed with signals recorded from multiple participants and multiple recording sessions; therefore, neural networks were more prone to learn optimal inter-participant and inter-session features to generalize properly. Conversely, during WS/CS trainings, neural networks were more prone to learn optimal session-specific/participant-specific features. Thus, representations associated with LOSO models were more likely to visualize general task-relevant spatio-temporal features, while those related to WS/CS models were more likely to include also session-specific/participant-specific and task-irrelevant features.

Saliency maps were computed for each deviant trial (containing the P300 response) belonging to each participant- and session-specific test set. Then, these maps were averaged across trials and folds (only for datasets 2 and 3), obtaining an average participant-specific and session-specific representation, named "spatio-temporal representation". Then, by averaging spatio-temporal representations across sessions (7 sessions for dataset 1 and 1 session for datasets 2 and 3), a participant-specific representation was computed, then normalized between $[-1,1]$ and finally averaged across participants, resulting in a "grand average (GA) spatio-temporal representation". This representation could be useful to study similarities between the temporal course of the gradients related to more relevant electrodes and the grand average ERPs of those specific electrodes. Additionally, the absolute value of each saliency map was also computed, and the absolute saliency maps were then averaged across trials, folds (only for datasets 2 and 3) and either the spatial or the temporal dimension to obtain an "absolute temporal or spatial representation", respectively, for each participant and session. Then, by averaging the absolute temporal/spatial representations across sessions, a participant-specific representation was computed, then normalized between $[0,1]$ and finally averaged across participants, resulting in a "GA absolute temporal or spatial representation", respectively. These absolute representations allowed the evaluation of the more class-discriminative time samples and electrodes for the P300 class, respectively.

4.2.7. Statistics

Before performing the statistical analyses, AUCs were computed for each participant- and session-specific test set and then averaged across sessions (7 sessions for dataset 1 and 1 session for datasets 2 and 3), in order to compare the performance metric at the level of participant. The following statistical comparisons were performed on the performance metric.

i. Pairwise comparisons between the MS-EEGNet and SOA algorithms (EEGNet, BranchedNet, OCLNN, xDAWN+RG) trained with WS, CS and LOSO strategies. AUCs were compared between the contrasted conditions separately for each dataset.

ii. Pairwise comparisons between the baseline MS-EEGNet and each of its variants, trained with the WS strategy. AUCs were merged together across different datasets and compared between the contrasted conditions using CNNs trained with the WS strategy; a similar procedure was adopted in [26,31] in order to evaluate the overall effect of the hyper-parameters of interest with the post-hoc evaluation.
iii. Pairwise comparisons between MS-EEGNet trained with the WS and TL-WS strategy, for each percentage of training examples of the new user and for each number of participants (M) from whom the knowledge was transferred to the new user (see Section 4.2.5-iv). This test was performed in order to evaluate the effect of the TL-WS strategy on the performance, as a function of the percentage of training examples and M. In these pairwise comparisons, AUCs were compared between the contrasted conditions, separately for each dataset.

The statistical analysis performed was the same used in [26,31]. In particular, Wilcoxon signed-rank tests were used to check for statistically significant differences between the contrasted conditions. To correct for multiple tests, a false discovery rate correction at $\alpha = 0.05$ using the Benjamini-Hochberg procedure [45] was applied.

4.3. RESULTS

4.3.1. Performance

MS-EEGNet and state-of-the-art algorithms

Table 4.3 reports the AUCs at the participant level (mean ± standard error of the mean, SEM) obtained with MS-EEGNet and with SOA algorithms using WS, CS and LOSO strategies, together with the results of the performed statistical tests.

Table 4.3 – AUC (% mean±SEM) obtained with MS-EEGNet and the re-implemented SOA algorithms adopting the WS, CS and LOSO strategies. The results of the performed Wilcoxon signed-rank tests (see Section 4.2.7-i) are also reported (*p<0.05, **p<0.01, ***p<0.001, corrected for multiple tests). Within each column, the bold characters are used to denote the best performance among the tested algorithms.

Algorithm	**Dataset 1**			**Dataset 2**		**Dataset 3**	
	WS	CS	LOSO	WS	LOSO	WS	LOSO
MS-EEGNet	**83.52±1.67**	**86.38±1.60**	75.40±1.81	**89.60±1.73**	74.82±3.04	**92.63±1.77**	**86.09±1.88**
EEGNet	82.53±1.83 **	85.88±1.63 **	75.76±1.71	87.98±2.65	75.15±3.01	91.22±1.92 *	83.30±2.53
BranchedNet	77.43±1.65 ***	84.20±1.82 ***	**76.03±1.86**	83.34±2.12 ***	72.39±2.89	91.60±1.53	84.84±1.46
OCLNN	75.95±1.64 ***	81.28±1.65 ***	71.40±1.42 **	79.92±2.78 ***	**75.21±3.14**	89.01±2.03 ***	83.73±1.59
xDAWN+RG	79.17±1.43 ***	80.89±1.32 ***	67.05±1.71 **	82.63±2.07 ***	73.83±2.71	90.03±1.87 *	82.40±2.77

MS-EEGNet scored an AUC of 83.52±1.67%, 89.60±1.73% and 92.63±1.77%, respectively when using signals from datasets 1-3 adopting a WS strategy. The proposed architecture significantly outperformed all the tested SOA algorithms when using dataset 1, while significantly outperformed BranchedNet, OCLNN and xDAWN+RG with dataset 2, and EEGNet, OCLNN and xDAWN+RG with dataset 3. In addition, adopting a CS strategy MS-EEGNet confirmed its decoding improvement respect to the SOA, scoring an AUC of 86.38±1.60% outperforming significantly all SOA algorithms. Lastly, adopting a LOSO strategy, MS-EEGNet scored an AUC of 75.40±1.81%, 74.82±3.04% and 86.09±1.88%, respectively when using signals from datasets 1-3. In this strategy, the proposed solution did not perform significantly better than other SOA solutions (see Section 4.4.1) except for dataset 1 where MS-EEGNet outperformed OCLNN and xDAWN+RG.

Design choices of MS-EEGNet

In the post-hoc hyper-parameter evaluation, we investigated the effect of particular design aspects of MS-EEGNet on the decoding performance, by statistically evaluating the difference in the AUCs between each variant MS-EEGNet and the baseline MS-EEGNet ($\Delta_{AUC} = AUC_{variant} - AUC_{baseline}$). The results are reported in Figure 4.2. In particular, the adoption of $N_b = 1$ (large), $N_b = 1$ (short), ${K_1}^{MST} = 8$, ${K_1}^{MST} = 16$ significantly worsened the performance, with an average drop in performance of 1.28, 3.46, 3.51, 1.72%.

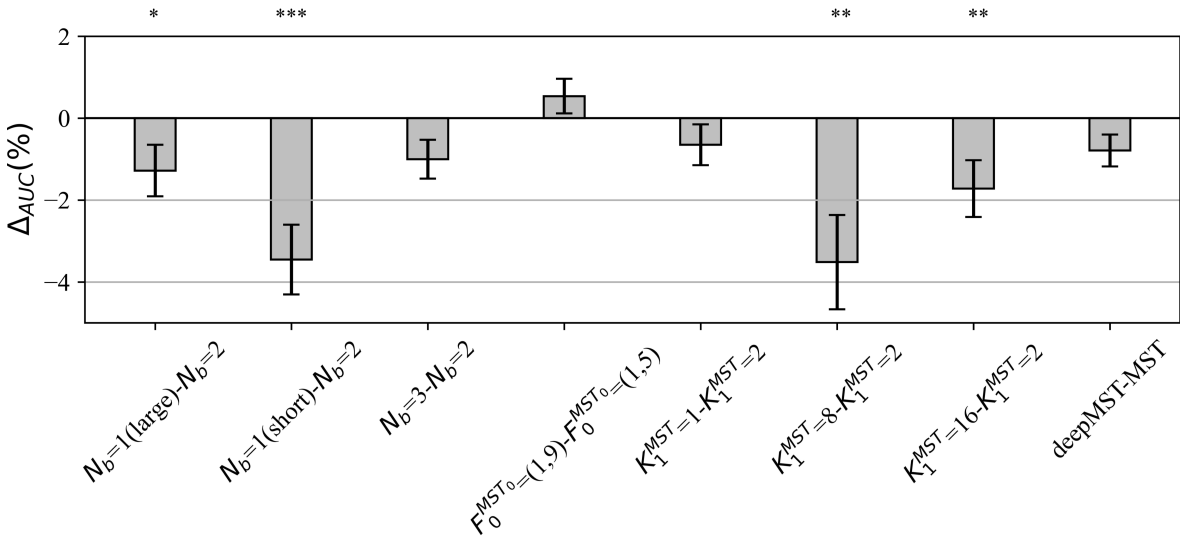


Figure 4.2 – Impact of alternative design choices of MS-EEGNet on the performance metric. The figure reports the difference between the AUC scored with the variant and with the baseline design (i.e., $\Delta_{AUC} = AUC_{variant} - AUC_{baseline}$) for each condition of the hyper-parameter (HP) tested, reported on the x-axis as "$HP_{variant} - HP_{baseline}$". The height of each grey bar represents the mean value across participants of Δ_{AUC}, while the error bar (black lines) represents the standard error of the mean. The results of the Wilcoxon signed-rank tests (see Section 4.2.7-ii) are also reported (*p<0.05, **p<0.01, ***p<0.001, corrected for multiple tests) on top of the figure.

Variable number of training examples: within-session and transfer learning strategies

The performance obtained with MS-EEGNet in the WS strategy as a function of the percentage of training examples (reported on the x-axis) is reported in Figures 4.3A-C (white bars), respectively for datasets 1-3. In all datasets, a percentage of training trials of 30%-45% was sufficient to obtain performance only a few points below that obtained with the entire training set, and in particular close or above 80%.

In addition, the performance obtained with MS-EEGNet in the TL-WS strategy is also reported, as a function of: i) the number of participants (M) adopted to design the LOSO-M model (grey and hatched bars); ii) the percentage of training examples. Lastly, the AUC difference between the TL-WS strategy and the WS strategy using the same percentage of training examples is shown in the lower panels of Figures 4.3A-C ($\Delta_{AUC} = AUC_{TL-WS} - AUC_{WS}$).

In case of dataset 1, the TL-WS strategy provided higher performance compared to the WS strategy (see the distributions of Δ_{AUC} reported in Figure 4.3) for each percentage of training examples $\forall M$. This occurred also in case of dataset 3 except in a couple of conditions ($M = 2$ using 30% and 60% of the training examples of the held back participant). Using dataset 2, TL was found beneficial only with the lowest number of training examples (i.e., 15%) $\forall M$ and using 60% of training examples with $M = 4$.

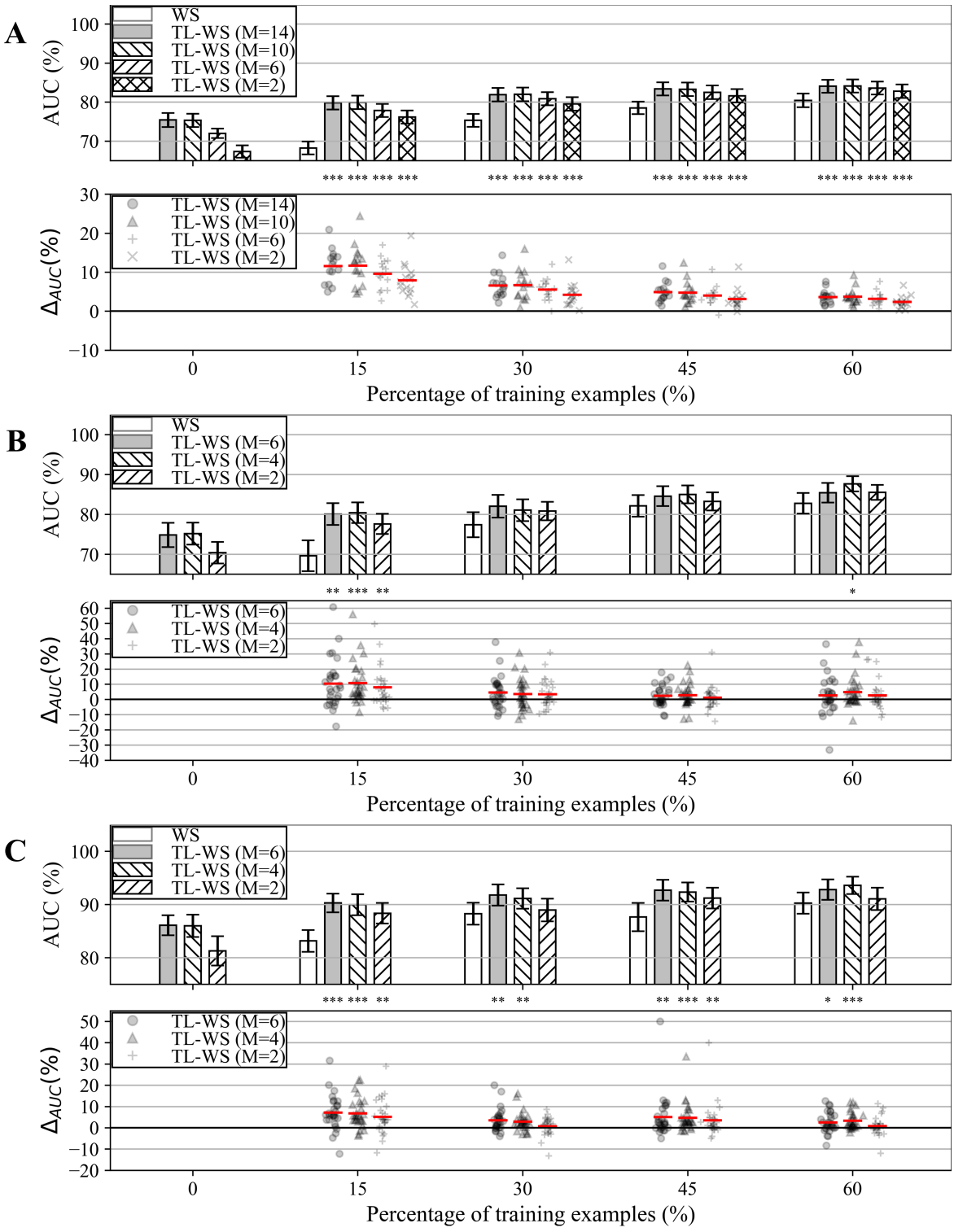


Figure 4.3 – Panels from A to C: AUC obtained with MS-EEGNet trained with the WS and TL-WS strategies for datasets 1-3. Top plot in each panel: The AUC obtained in WS (white bars) is reported as a function of the percentage of training examples (reported on the x-axis), while the AUC obtained in TL-WS is reported also as a function of the number of participants (M) used to optimize the LOSO-M models (grey and hatched bars). The height of each bar represents the mean value of the performance metric across participants, while the error bar (black lines) represents the standard error of the mean. Bottom plot in each panel: The AUC difference between the TL-WS and WS strategy (i.e., $\Delta_{AUC} = AUC_{TL-WS} - AUC_{WS}$) using the same percentage of training examples is reported using markers and a red line denotes the mean value. For each percentage, a Wilcoxon signed-rank test was performed (see Section 4.2.7-iii) to compare TL-WS vs. WS strategy, and the statistical significance is reported (*p<0.05, **p<0.01, ***p<0.001, corrected for multiple tests) on top of each plot.

4.3.2. Explaining P300 decision: gradient-based representations

In this section, we analyze the features of the input variables that most strongly supported the P300 classification decision in MS-EEGNet.

Spatio-temporal representations

The top panels of Figure 4.4 (Figures 4.4A-C) display the grand average spatio-temporal representation of MS-EEGNet trained with the LOSO strategy using signals from datasets 1-3. From these figures, the more class-discriminative electrodes can be identified, i.e., P4, Pz and CP1 for datasets 1-3, respectively. The grand average ERPs for the standard and deviant stimuli of these representative electrodes are displayed in the lower panels of Figure 4.4 (Figures 4.4D-F).

In case of dataset 1, P4 appeared as the most important electrode, in particular from 300 to 550 ms. Three main peaks can be identified: two positives at 350 and 510 ms, and one negative at approximately 410 ms (Figure 4.4A). These peaks correspond to the peaks in the grand average ERP of the deviant stimulus at approximately the same times (Figure 4.4D). In case of datasets 2 and 3, the most important site was Pz from 300 to 400 ms and CP1 from 350 to 400 ms, respectively. In these cases, a single positive peak occurred in the spatio-temporal maps at about 350 and 390 ms, respectively (Figures 4.4B and 4.4C) and was associated to the peak in the grand average ERP of the deviant stimuli at approximately the same time (Figures 4.4E and 4.4F).

In the following sections the interpretation of the relevant input features driving MS-EEGNet P300 decision are analyzed separately in the temporal and spatial domains.

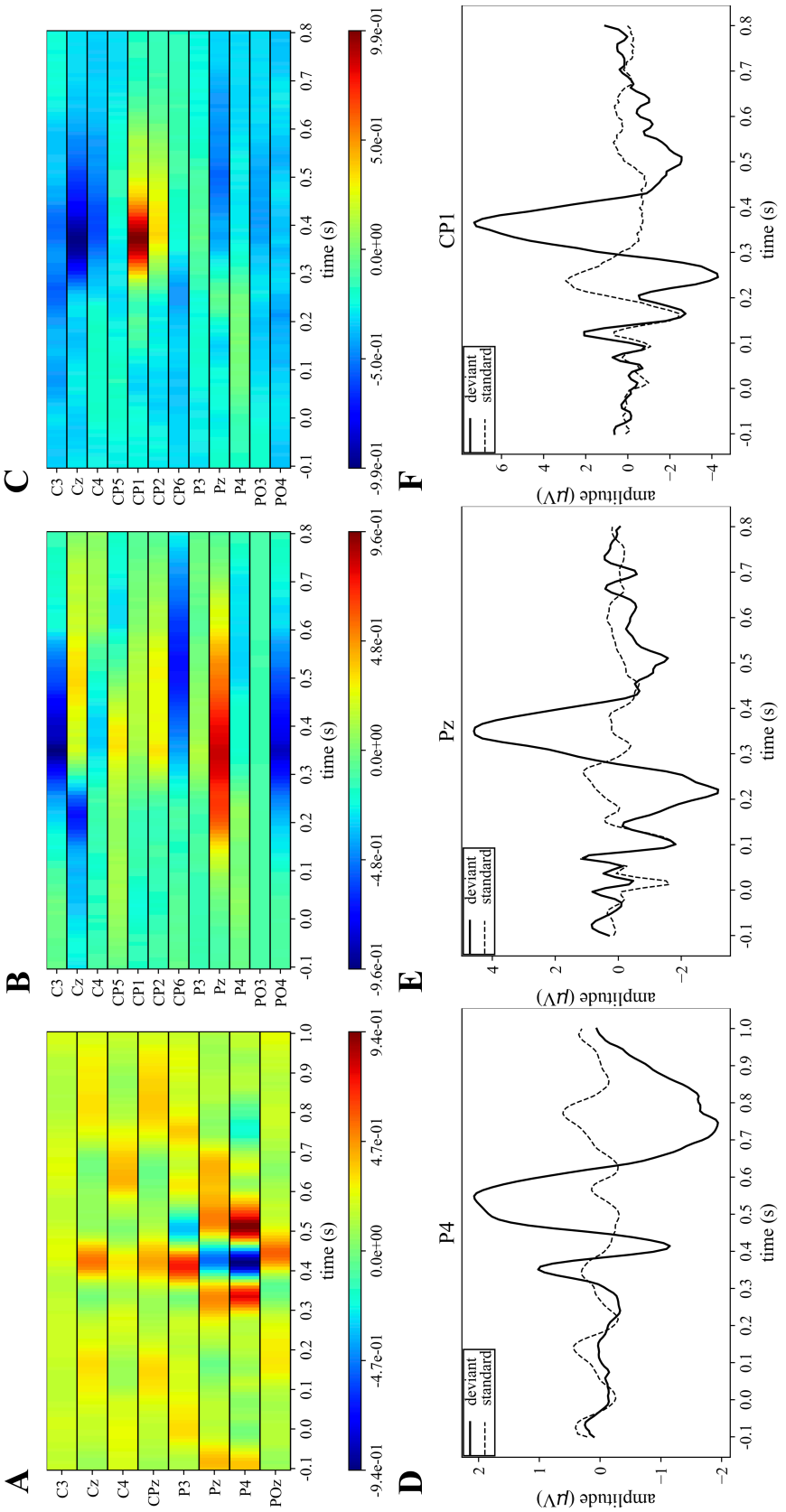


Figure 4.4 – Grand average spatio-temporal representations. The top panels (Figures 4.4A-C) show the grand average spatio-temporal representation of MS-EEGNet trained with the LOSO strategy using signals from datasets 1-3. Positive gradients are shown in red, while negative gradients in blue. The bottom panels (Figures 4.4D-F) show the grand average ERP for the deviant (black lines) and standard (dashed black lines) stimuli associated to the most relevant electrode (the one with the largest gradient values) for datasets 1-3.

Absolute temporal representations

Figure 4.5 displays the grand average absolute temporal representations of MS-EEGNet trained with the LOSO strategy using signals from datasets 1-3 (Figures 4.5A-C). These patterns highlight, by means of local and global peaks, the more class-discriminative time samples for the P300 class across all spatial sites. These waveforms confirm the highest importance of time samples approximately between 300 and 550 ms in all cases, with the peak at about 410 ms, 350 ms and 390 ms for datasets 1-3, respectively, in agreement with results in Figure 4.4. Interestingly, these waveforms synthetically highlight how the network learns different temporal profiles of sample relevance depending on the dataset, e.g., more regular waveforms in case of dataset 2 and 3 (but more spiking in case of dataset 3) and more irregular waveform in case of dataset 1 (with several local maxima, two in particular just next the global one, i.e., at 350 and 510 ms). These differences may be linked to the different sensory modalities involved (visual vs. auditory) or to the different participants (healthy vs. pathological), or to the different paradigms used to elicit P300 (oddball paradigm vs. flashing the object under fixation).

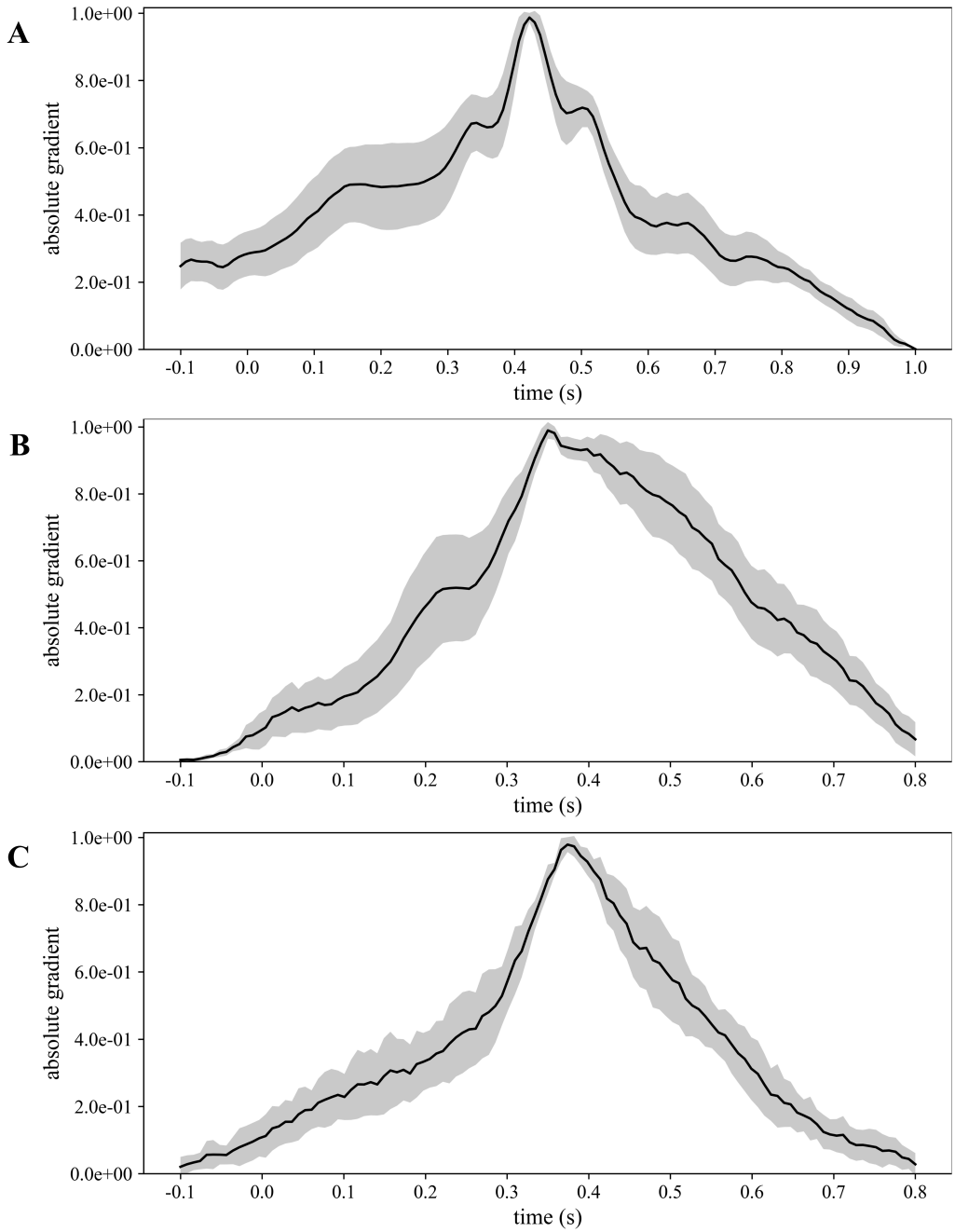


Figure 4.5 – Grand average absolute temporal representations of MS-EEGNet trained with the LOSO strategy using signals from datasets 1-3 (Figures 4.5A-C); the mean value (black line) ± standard deviation (grey shaded areas) across participants is represented.

Absolute spatial representations

Besides the investigation of the more P300-discriminative temporal features, it is also interesting to evidence the more P300-discriminative spatial features. To this aim, Figure 4.6 shows the grand average absolute spatial representations of MS-EEGNet trained with the LOSO strategy using signals from datasets 1-3 (Figures 4.6A-C), emphasizing the different spatial profiles of the sample relevance.

The three more class-discriminative electrode sites across all time samples were (in increasing order of relevance) Pz, P3 and P4 when the CNN was trained on dataset 1; C3, Cz and Pz when

the CNN was trained on dataset 2; Cz, CP2 and CP1 when the CNN was trained on dataset 3. Again, these differences can be associated to differences in the sensory modality, participants, paradigms adopted across the three datasets.

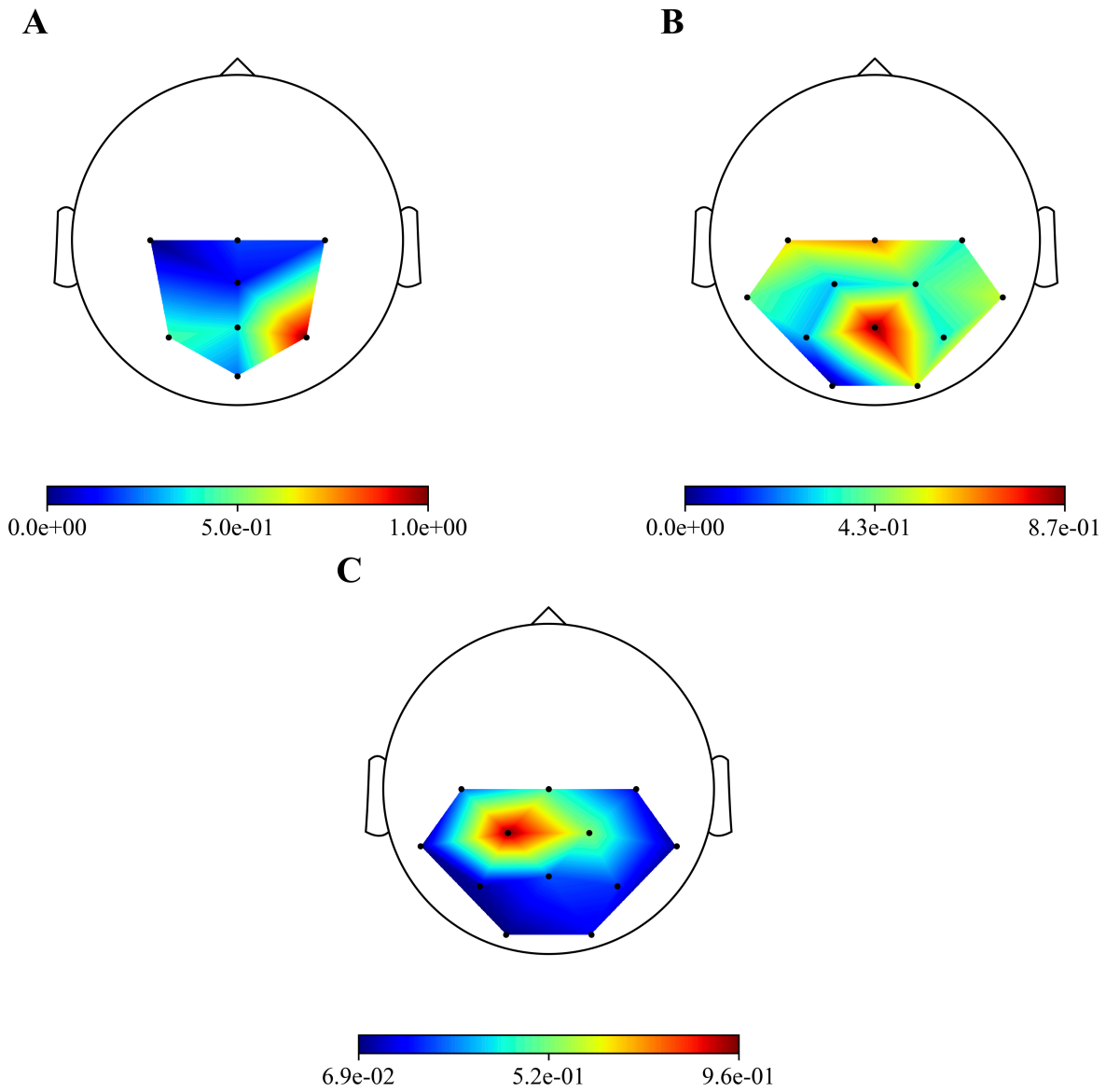


Figure 4.6 – Grand average absolute spatial representations of MS-EEGNet trained with the LOSO strategy using signals from datasets 1-3 (Figures 4.6A-C).

Progressive changes of spatio-temporal sample relevance while increasing training examples

Lastly, the absolute temporal and spatial representations were also used to analyze the progressive change in the importance of the spatio-temporal samples while increasing the percentage of training examples included when training MS-EEGNet with TL-WS and WS strategies. For the TL-WS condition, only CNNs initialized from LOSO models with the largest number of participants were considered. The absolute temporal and spatial representations are reported in Figure 4.7, in case of a representative participant and session belonging to dataset 1.

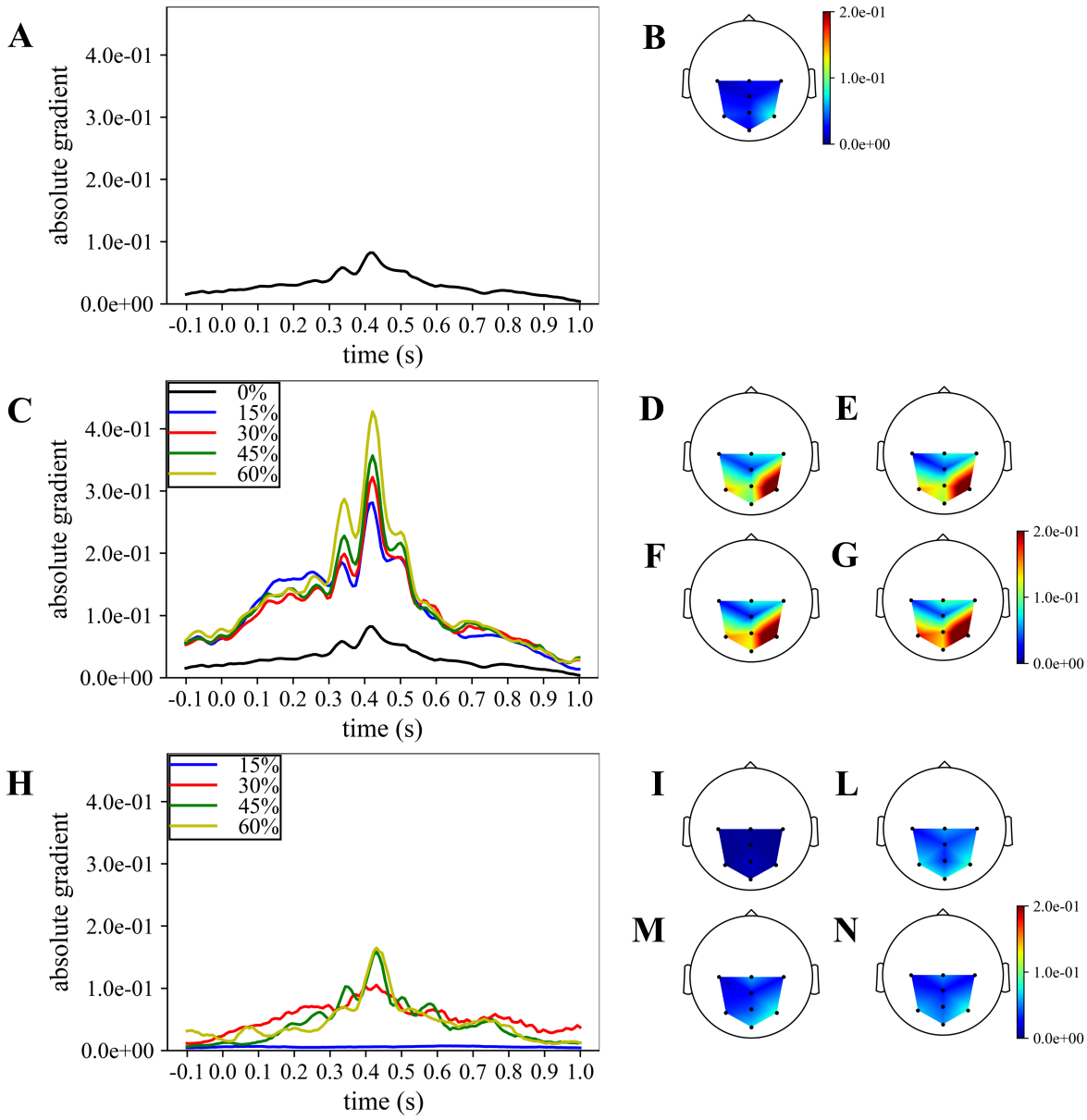


Figure 4.7 –Grand average temporal and spatial absolute representations of MS-EEGNet trained on the dataset 1 for a representative participant and session, adopting the LOSO, TL-WS and WS strategies. In particular, the representations obtained with the LOSO strategy in the temporal and spatial domains are reported in Figures 4.7A and 7B, respectively. The representations obtained with the TL-WS strategy in the temporal and spatial domains are reported, respectively, in Figure 4.7C (coloured lines) and Figures 4.7D-G, as the percentage of training examples of the new participant increased (15, 30, 45, 60%, from Figure 4.7D to 4.7G, respectively). The representations obtained with the WS strategy in the temporal and spatial domains are reported, respectively, in Figure 4.7H (coloured lines) and Figures 4.7I-N, as the percentage of training examples of the participant increased (15, 30, 45, 60%, from Figure 4.7I to 4.7N, respectively). Note that in order to maintain the same scale across strategies in the spatial absolute representations, in Figures 4.7D-G the maximum gradient value represented (2.0e-1) was below the real maximum gradient value (3.3e-1), saturating the value in particular around P4.

In particular, Figures 4.7A and 4.7B report the absolute temporal and spatial representations as obtained in the LOSO strategy. Figures 4.7C-G show the effects of TL-WS strategy, as the percentage of training examples from the held back participant increased. While transferring the knowledge from other participants and sessions, the CNN inherited the importance profile

from the pre-trained condition. Thus, for each percentage of training examples (Figures 4.7C-G) the temporal and spatial profiles did not change substantially their shape from the LOSO condition since the importance in the temporal and spatial domains was already learned in the LOSO training. Nevertheless, the amplitude increased both in the temporal and spatial domains while increasing the percentage of training examples, indicating a progressive accumulation of the importance. Conversely, adopting a WS strategy (Figures 4.7H-N), the CNN was randomly initialized and, therefore, the CNN had to learn from scratch the more class-discriminative spatio-temporal samples. Thus, the temporal and spatial profiles changed more respect to TL-WS as the percentage of training examples increased. In particular, temporal profiles changed from a nearly flat profile (e.g., 15% in Figure 4.7H) to profiles more focused on time samples in the range 300-550 ms (e.g., 45, 60% in Figure 4.7H) peaking at approx. 410 ms. Furthermore, spatial profiles changed from a diffused distribution (Figure 4.7I) to distributions more focused on parietal electrodes (in particular P3, Pz and P4 in Figures 4.7L-N). However, the absolute gradients resulted lower than in the TL-WS condition, in particular in correspondence of the more class-discriminative temporal (i.e., 350, 410, 510 ms) and spatial (P3, Pz and P4) samples.

4.4. DISCUSSION

In this study, a lightweight multi-scale CNN design for EEG decoding named MS-EEGNet was proposed and applied to decode the P300 event from 3 different datasets. This CNN merges the multi-scale temporal learning proposed by Farahat et al. [16], with the lightweight characteristics originally proposed in EEGNet [18], operating even a further decrease in the number of trainable parameters while learning multi-scale features. MS-EEGNet was compared with many SOA algorithms, including CNNs (EEGNet, BranchedNet, OCLNN) and a traditional ML pipeline (xDAWN+RG). To better analyze the multi-scale feature learning as operated by MS-EEGNet, we performed a post-hoc analysis on the hyper-parameters. In addition, MS-EEGNet was extensively evaluated in 4 training conditions, each one reflecting a different practical scenario: i) using participant-specific signals of single recording sessions (WS); ii) using participant-specific signals of multiple recording sessions (CS); iii) using signals from other participants (LOSO); iv) using a fraction of participant-specific signals from a pre-trained cross-participant CNN (TL-WS). Lastly, we exploited saliency maps to obtain representations aimed to explain MS-EEGNet decision by visualizing the relevant samples in the input domain. Both the proposed architecture and the performed analyses represent significant expansion compared to our previous work [20], limited to the application of a design based on EEGNet to solve the P300 task proposed by IFMBE 2019 scientific challenge (corresponding to dataset 1 here). In the following, the performance of MS-EEGNet and the results of the performed analyses are critically discussed.

4.4.1. Performance of MS-EEGNet and comparison with state-of-the-art algorithms

The performance of MS-EEGNet using WS strategy was above 80% for all datasets, reaching higher values for datasets 2 and 3 compared with dataset 1 (Table 4.3). This difference could depend on several factors such as different paradigms, stimuli and populations (ASD vs. healthy), possibly leading to different P300 responses, e.g., with lower or larger amplitude. Regarding to this, Figures 4D-F show that the P300 response to the deviant stimulus in the dataset 1was indeed characterized by a lower amplitude, perhaps increasing the difficulty in discriminating between standard/deviant stimuli. Other contributing factors could be the lower proportion between training and test examples, and the lower number of electrodes in dataset 1 vs. datasets 2 and 3. It is worth noticing that this same difference in WS performance across the datasets was notable in the other algorithms too. Using the CS strategy, the performance improved compared to the WS strategy for all algorithms and this result is in line with [21]. When comparing MS-EEGNet to the other algorithms, our design exhibited the highest performance on each dataset, adopting the WS and CS strategies. Interestingly, among the tested CNNs, OCLNN (that uses a mixed spatio-temporal convolution), and BranchedNet (that performs a spatial convolution first) performed generally lower than MS-EEGNet and EEGNet (that perform temporal convolution first). This is in line with [21], where our previous design adapted from EEGNet outperformed significantly a CNN design inspired from Manor and Geva [14] that used a first spatial convolutional layer. Therefore, these results suggest that a CNN design trained on participant-specific signals and based on a first temporal filtering of EEG signals, leads to higher P300 decoding performance than other solutions that use a first mixed spatio-temporal or a first spatial filtering of the input signals. Hence, higher performance

could be achieved learning temporal features directly from raw EEG signals (exploiting useful raw temporal information related to the P300 event) instead from signals with a higher level of abstraction. Overall, among the tested SOA CNNs, EEGNet is the one exhibiting the closest performance to MS-EEGNet; this can be explained by the derivation of MS-EEGNet from EEGNet with the addition of multi-scale temporal feature learning and compressed representation learning. However, the results denote that the changes included in MS-EEGNet can significantly improve the high performance already achieved by EEGNet - especially using session-specific (WS) and participant-specific (CS) input distributions, see datasets 1 (p=2e-3 and p=3e-3 with WS and CS strategies, respectively) and 3 (p=4e-2) in Table 4.3 - using a lower number of trainable parameters.

As expected, adopting the LOSO strategy caused an overall drop of the performance metric across all the tested approaches, respect to WS and CS strategies; the different approaches generally provided similar performance (MS-EEGNet only performed significantly better than xDAWN+RG and OCLNN in dataset 1).

Hence, overall, MS-EEGNet performed better than the other SOA algorithms in WS and CS strategies and behaved similar as other SOA algorithms in the LOSO strategy. This becomes more relevant considering that MS-EEGNet is the lightest CNN among the tested ones, as EEGNet, BranchedNet and OCLNN introduced more trainable parameters (see Table 4.2). Indeed, this is particularly important as in practice it is common to deal with small EEG datasets; thus, keeping limited the number of trainable parameters is crucial when designing CNNs for EEG decoding, in order to avoid overfitting. Likely, the lightweight design of MS-EEGNet may explain the absence of higher performance in LOSO strategy, due to the peculiarities of LOSO training. In this case, class-discriminative features are learned from input distributions with very large variability, involving different participants and possibly different sessions (e.g., with dataset 1). Thus, the CNN, besides needing more training examples, may need more capacity (i.e., more layers/more parameters) to solve the task with higher performance. Considering that our CNN is the lightest among the tested ones (see Table 4.2), obtaining performance similar than other CNNs should not be surprising (and rather can be still considered a satisfactory result). In LOSO strategy, MS-EEGNet significantly outperformed the traditional ML approach only for dataset 1. This may indicate that in a LOSO strategy MS-EEGNet can learn more relevant cross-participant features, leading to significant higher performance, than a ML pipeline when a larger dataset is used, as in case of dataset 1. Lastly, besides performance and parameters to fit, considerations about the training time are relevant for practical usage. The multi-scale SOA CNN (BranchedNet) resulted slower to train respect to MS-EEGNet, while single-scale SOA CNNs (EEGNet and OCLNN) were faster to train. Overall, compared to SOA CNNs, MS-EEGNet represented a good compromise between performance, model size and computational time.

4.4.2. Performance of MS-EEGNet: post-hoc hyperparameter evaluation

We performed a post-hoc hyper-parameter evaluation of 8 variant design choices of MS-EEGNet by varying 4 different hyper-parameters of the multi-scale temporal block (Figure 4.2). Using a single-scale variant ($N_b = 1$) including only the large- or the short-scale a reduction in trainable parameters and in training time was observed respect to the baseline MS-

EEGNet (see Table 4.2). At the same time, the performance significantly worsened in both cases, indicating the benefit of the multi-scale temporal feature learning respect to single-scale feature learning for P300 decoding, at the expense of an increased number of trainable parameters and computational time. In addition, the different impact on the performance observed in the design $N_b = 1$ (large) and $N_b = 1$ (short) suggests that the temporal features learned in the large-scale branch were more class-discriminative. Interestingly, using one additional intermediate time scale (3-branched variant $N_b = 3$) a non-significant difference in performance was observed compared to the baseline MS-EEGNet while more parameters and training time were required (see Table 4.2). These results about the number of branches of MS-EEGNet suggest that the dual-branched design represented a good compromise between performance, model size and training time.

Furthermore, the alternative ratio $r^{MST} = {}^{1}/_{2}$ between the two time scales obtained with ${F_0}^{MST_0} = (1{,}9)$ (corresponding to learning summaries of about 500 ms and 250 ms), resulted in a small not significant (p=0.06) increase in performance respect to the baseline MS-EEGNet ($r^{MST} = {}^{1}/_{4}$), requiring few more parameters and training time. In addition, variants learning more feature maps (${K_1}^{MST} = 8$ and ${K_1}^{MST} = 16$), respect to the compressed representation exploited in the baseline MS-EEGNet (${K_1}^{MST} = 2$), not only required more parameters to fit and were slower (see Table 4.2), but worsened the performance significantly. This suggest that learning compressed representations could be beneficial – both in terms of performance, model size and training time – for P300 decoding. Remarkably, the variant architecture including the most extreme compressed representation (${K_1}^{MST} = 1$), i.e., learning only one feature map for each time scale, scored similar performance as the baseline MS-EEGNet, while lightly reducing the model size and requiring the same training time (see Table 4.2), suggesting that future architectures could exploit also this design to further reduce the model size without hampering the performance. Lastly, increasing the depth of the MST block, did not provide any significant improvement in performance, introduced more parameters to fit and required more training time (see Table 4.2). Thus, these last results suggest that a shallower and lightweight MST design, as provided in the baseline MS-EEGNet, is preferable for P300 decoding.

4.4.3. Performance of MS -EEGNet: transfer learning strategy and variable number of training trials

MS-EEGNet was capable to deal with a reduced number of training trials when trained from scratch (WS), although not at the smallest percentage of training trials (Figure 4.3). The performance increased in TL-WS. Indeed, transferring the knowledge using the smallest percentage of training examples of the held back participant (i.e., 15%) resulted beneficial, compared to WS, across all datasets and regardless the number of participants from whom the knowledge was transferred (Figure 4.3). This beneficial effect of the TL-WS strategy was found also when using more training examples (30, 45, 60%) of the held back participant on datasets 1 and 3. As expected, the worst performance was obtained when transferring the knowledge from LOSO models trained on the smallest subset of participants ($M = 2$) for all datasets and percentages; however, this condition produced a significant increase in

performance compared to randomly initialized models especially when using a small amount of signals belonging to the new user (i.e., 15%). Therefore, pre-trained models do not necessarily need to be optimized on a large set of participants in order to significantly outperform randomly initialized models, especially when using a small amount of data during transfer learning (see also Section 4.6.3 in Supplementary Materials for comparison between TL-WS and WS with 100% of training trials).

Overall, these results suggest that the proposed approach could be used to accurately decode the P300 event even with a reduced number of standard/deviant stimuli presented to the user during the calibration stage.

4.4.4. Explaining P300 decision

The proposed approach achieved high performance outperforming SOA algorithms. As stated by Montavon et al. [22], in practice it is also crucial to verify that the decoding performance results from a proper problem representation and not from the exploitation of artefacts in the input data. Therefore, in the present study we explained MS-EEGNet decision for P300 decoding via saliency maps, providing the GA spatio-temporal, GA absolute temporal and GA absolute spatial representations of the relevance of the input samples.

The GA spatio-temporal representations of MS-EEGNet (Figures 4.4A-C) evidenced higher values (both positive and negative) of the gradients, corresponding to more class-discriminative input samples, within time intervals (roughly between 300-550 ms) matching the P300 temporal occurrence for all datasets. The positive/negative peaks in these gradient patterns corresponded to peaks in the GA ERPs of the deviant stimulus (Figures 4.4D-F). Indicating with i and j the row and column indices, respectively, positive and negative gradients in the (i,j) location in Figures 4.4A-C represent the direction in which a change of the (i,j) input feature increased the P300 class score and, consequently, the CNN decision toward the P300 class. Thus, for example analyzing the gradients related to P4 obtained from dataset 1 (Figure 4.4A), two positive peaks and one negative peak were found. As the P4 input signal of a deviant trial increased its value at the two positive peaks (at about 350 and 510 ms), the deviant condition differed more than the standard condition, resulting easier to distinguish the deviant class and providing a higher score to it. Therefore, these peaks in the deviant GA ERP were associated with positive gradient peaks. Conversely, as the P4 input signal of a deviant trial reduced its value at the local minimum (at ~410 ms), the negative peak resulted more distant from the standard condition, leading to a higher score for the deviant class (negative gradient peak). This consideration can be extended to datasets 2 and 3, by analyzing Pz and CP1 electrodes respectively. Therefore, as already obtained in [16], higher differences in the ERP between deviant and standard stimuli are reflected onto the saliency maps by means of positive and negative gradients.

When computing the absolute value of the saliency maps, the absolute gradient at the spatio-temporal sample (i,j) reflects how much a change in this sample affects the P300 class score. We analyzed the absolute saliency maps separately in the time- and spatial-domains (Figures 4.5 and 4.6), in order to evidence the more discriminative temporal samples and electrodes, respectively, independently on the direction (positive or negative) they contributed to the decoding result. The GA temporal absolute profile for each dataset peaked approximately in

correspondence of the peak of the P300 response. Interestingly, the absolute temporal representations exhibit different patterns for the three datasets, evidencing that they are able to detect differences embedded in the P300 response across the three datasets. Lastly, the GA absolute spatial distributions represented in a topological map allowed a direct analysis of the more P300 discriminative electrodes of MS-EEGNet. These were mainly distributed in the parietal area and centro-parietal area. This may provide practical hints to reduce the number of electrodes in the design of P300-BCIs. Overall, the various gradient-based representations (Figures 4.4-4.6) matched the P300 spatio-temporal distribution, confirming that MS-EEGNet was able to capture meaningful task-related features, without exploiting artefactual/noisy input sources.

Interestingly, by using a representative example, we show that while transferring the knowledge the importance of temporal and spatial samples gradually increased from the LOSO condition (Figures 4.7A and 4.7B) as the percentage of training examples increased. In particular, it appears that the more task-relevant temporal and spatial samples were already learned in the LOSO strategy; however, during transfer learning (Figures 4.7C-4.7G) the LOSO temporal and spatial profiles (template profiles) were modelled on the new participant- and session-specific training distribution, giving progressively more importance to particular temporal intervals/electrode sites starting from the template profiles. The availability of these template profiles allowed a rapid learning of the relevant participant-specific and session-specific input samples (i.e., needing a low number of training examples of the new participant). Conversely, when training CNNs from scratch with the WS strategy, the profile distribution rapidly changed its shape both in the temporal (Figure 4.7H) and spatial (Figures 4.7I-N) domains but reached lower importance values compared to TL-WS strategy. When transferring the knowledge, the profile was more focused on the interval 300-550 ms with three distinct main peaks and on the sites P4>P3>Pz already at the lowest percentage (15%, Figures 4.7C and 4.7D), while at the same percentage, the WS strategy was characterized by more flat and homogeneous distributions (Figures 4.7H and 4.7I). These considerations could explain the performance improvement obtained in the TL-WS strategy (Figure 4.3): the parameters learned using the LOSO strategy overall represented a better initialization point in the parameter space compared to a random one.

4.5. CONCLUSIONS

In conclusion, we wish to stress that the present study aims to contribute to uncovering the enormous potentialities of deep learning via CNNs for EEG decoding, and to their exploitation in practice adopting different training strategies, reflecting different scenarios. Our multi-scale design was the most lightweight, at the same time outperforming many SOA algorithms when using three different P300 datasets, indicating that care has to be taken to design CNNs for EEG decoding keeping limited the parameters to fit, especially when handling small datasets (not as large as the ones adopted in the computer vision field, e.g., >100K of examples). In addition, the hyper-parameter post-hoc analysis confirmed that the innovative aspects of our architecture, i.e., the design of a lightweight multi-scale temporal block implemented via separable convolutions and the use of compressed representation learning, were beneficial. Crucially, the capability of MS-EEGNet to transfer the knowledge with high performance even with a small amount of training examples, could be highly useful in practice to reduce the calibration time of P300-based BCIs on a new user.

Saliency maps confirmed their utility to explain the neural network decision in P300 decoding tasks; the derived spatial and temporal representations resulted to match the P300 spatio-temporal distribution. However, the utility of these representations is not limited to provide an additional validation of the algorithm. Indeed, the CNN ability to learn automatically the most meaningful features to perform classification, gives the possibility to use these algorithms as data-driven EEG analysis tools; then, the use of the saliency maps (or similar representations) allows to interpret the CNN decision and it is possible to take advantage of these interpretations for increasing the comprehension of the brain dynamics underlying the decoded events (e.g., P300 response). For example, representations derived from saliency maps (in the time- and/or spatial-domain), could be used to study the variability between participants (i.e., which features of the input samples are more/less consistent across participants) and within-participant (i.e., by comparing representations associated to early and late trials, e.g., to investigate the effects of training or treatment). Furthermore, the analysis of between-participants and within-participant variabilities could be useful, in perspective, to develop biomarkers to diagnose and monitor neurological or psychiatric disorders [16], e.g., P300 amplitude, latency and topographical alterations in mild cognitive impairment [46], dementia [47] and schizophrenia [48]. In addition, identifying the more class-discriminative temporal and spatial input features can also have a relevant practical impact in the design of BCIs. For example, the identification of a small subset of more relevant electrodes (as we found here) may drive the definition of BCI systems with a very small electrode montage, increasing the comfort of the participant and reducing the preparation time. It is worth noticing that performing this analysis on within-participant CNNs, the optimal electrode montage could also be identified on an individual basis.

Overall, the present study, by specifically addressing the aspects of lightweight design, transfer learning and interpretability of the proposed CNN, can contribute to advance the development of deep learning-based decoders for P300-BCIs. Future developments include the application of the proposed architecture to other ERP decoding tasks, and the adoption of interpretable and more lightweight layers such as the sinc-convolutional layer to perform band-pass filtering [25,26,49]. In addition, automatic hyper-parameter search [50] will be exploited

to further improve MS-EEGNet design and other explanation techniques such as layer-wise relevance propagation will be investigated, carefully analyzing the effect of different propagation rules and parameters for EEG decoding.

4.6. SUPPLEMENTARY MATERIALS

4.6.1. Details about the state-of-the-art CNNs

The main details of the architectures considered in this study are reported in the Supplementary Tables 4.1-4.3, respectively for our adaptation of EEGNet that won the IFMBE 2019 competition [20] (code and documentation available at https://github.com/ddavidebb/IFMBE2019Challenge-BCIAUT-P300), BranchedNet [16] and OCLNN [17]. Since BranchedNet was designed to accept EEG signals sampled at 250 Hz as input, we changed the temporal pool sizes and kernel lengths accordingly (i.e., dividing them with a factor of 2), as done in previous studies when re-implementing CNNs on other datasets [18,26]. Codes of all the re-implemented CNNs are available together with the ones of MS-EEGNet at https://github.com/ddavidebb/P300_decoding_MS-EEGNet.

Supplementary Table 4.1 – Architecture details of the EEGNet adaptation used in [20]. Each layer is provided with its name, main hyper-parameters and number of trainable parameters. See Sections 4.2.1 and 4.2.2 of the manuscript for the meaning of the symbols. The total number of trainable parameters was 1386, when using signals from the dataset 1, and 1418, when using signals from datasets 2 and 3. In all layers, where not specified, stride (S) and padding (P) were set to $(1,1)$ and $(0,0)$, respectively.

Block	Layer name	Hyper-parameters	Number of trainable parameters
	Input	$K_0 = 1$	0
ST	Conv2D	${K_0}^{\mathrm{ST}} = 8$, ${F_0}^{\mathrm{ST}} = (1,65)$, ${P_0}^{\mathrm{ST}} = (0,32)$	${F_0}^{\mathrm{ST}}[0] \cdot {F_0}^{\mathrm{ST}}[1] \cdot {K_0}^{\mathrm{ST}} \cdot K_0$
	BatchNorm2D	$m = 0.99$	$2 \cdot {K_0}^{\mathrm{ST}}$
	Depthwise-Conv2D	${D_1}^{\mathrm{ST}} = 2$, ${K_1}^{\mathrm{ST}} = {K_0}^{\mathrm{ST}} \cdot {D_1}^{\mathrm{ST}}$, ${F_1}^{\mathrm{ST}} = (C, 1)$, kernel max norm=1	${F_1}^{\mathrm{ST}}[0] \cdot {F_1}^{\mathrm{ST}}[1] \cdot {K_1}^{\mathrm{ST}}$
	BatchNorm2D	$m = 0.99$	$2 \cdot {K_1}^{\mathrm{ST}}$
	ELU	$\alpha = 1$	0
	AvgPool2D	${F_p}^{\mathrm{ST}} = {S_p}^{\mathrm{ST}} = (1,4)$	0
	Dropout	$p = 0.25$ or $p = 0.5$	0
Temporal (T)	Separable-Conv2D	${D_0}^{T} = 1, {K_0}^{T} = {K_1}^{ST} \cdot {D_0}^{T}$, ${F_0}^{T} = (1,17)$, ${P_0}^{T} = (0,8)$	${F_0}^{T}[0] \cdot {F_0}^{T}[1] \cdot {K_0}^{T} + \left({K_0}^{T}\right)^2$
	BatchNorm2D	$m = 0.99$	$2 \cdot {K_0}^{T}$
	ELU	$\alpha = 1$	0
	AvgPool2D	${F_p}^{T} = {S_p}^{T} = (1,8)$	0
	Dropout	$p = 0.25$ or $p = 0.5$	0
FC	Flatten		0
	Fully-Connected	$N^{FC} = 2$, kernel max norm=0.25	$N^{FC} \cdot ({T_p}^{T} \cdot {K_0}^{T} + 1)$
	Softmax		0

Supplementary Table 4.2 – Architecture details of BranchedNet [16]. Each layer is provided with its name, main hyper-parameters and number of trainable parameters. See Sections 4.2.1 and 4.2.2 of the manuscript for the meaning of the symbols. The total number of trainable parameters was 5418, when using signals from the dataset 1, and 7954, when using signals from datasets 2 and 3. In all layers, where not specified, stride (S) and padding (P) were set to $(1,1)$ and $(0,0)$, respectively.

Block	Layer name	Hyper-parameters	Number of trainable parameters
	Input	$K_0 = 1$	0
Spatial (S)	Conv2D	$K_0{}^S = 12, F_0{}^S = (C, 1)$	$F_0{}^S[0] \cdot F_0{}^S[1] \cdot K_0{}^S \cdot K_0$
	BatchNorm2D	$m = 0.99$	$2 \cdot K_0{}^S$
	Tanh		0
	Dropout	$p = 0.25$ or $p = 0.5$	0
MST scale 0	Conv2D	$K_0{}^{MST_0} = 4, F_0{}^{MST_0} = (1,12)$	$F_0{}^{MST_0}[0] \cdot F_0{}^{MST_0}[1] \cdot K_0{}^{MST_0} \cdot K_0{}^S$
	BatchNorm2D	$m = 0.99$	$2 \cdot K_0{}^{MST_0}$
	Tanh		0
	Dropout	$p = 0.25$ or $p = 0.5$	0
	AvgPool2D	$F_p{}^{MST_0} = S_p{}^{MST_0} = (1,2)$	0
	Conv2D	$K_1{}^{MST_0} = 8, F_1{}^{MST_0} = (1,12)$	$F_1{}^{MST_0}[0] \cdot F_1{}^{MST_0}[1] \cdot K_1{}^{MST_0} \cdot K_0{}^{MST_0}$
	BatchNorm2D	$m = 0.99$	$2 \cdot K_1{}^{MST_0}$
	Tanh		0
	Dropout	$p = 0.25$ or $p = 0.5$	0
	Conv2D	$K_2{}^{MST_0} = 8, F_2{}^{MST_0} = (1,12)$	$F_2{}^{MST_0}[0] \cdot F_2{}^{MST_0}[1] \cdot K_2{}^{MST_0} \cdot K_1{}^{MST_0}$
	BatchNorm2D	$m = 0.99$	$2 \cdot K_2{}^{MST_0}$
	Tanh		0
	Dropout	$p = 0.25$ or $p = 0.5$	0
	Flatten		0
MST scale 1	Conv2D	$K_0{}^{MST_1} = 4, F_0{}^{MST_1} = (1,52)$	$F_0{}^{MST_1}[0] \cdot F_0{}^{MST_1}[1] \cdot K_0{}^{MST_1} \cdot K_0{}^S$
	BatchNorm2D	$m = 0.99$	$2 \cdot K_0{}^{MST_1}$
	Tanh		0
	Dropout	$p = 0.25$ or $p = 0.5$	0
	AvgPool2D	$F_p{}^{MST_1} = S_p{}^{MST_1} = (1,2)$	0
	Flatten		0
FC	Concatenate		0
	Fully-Connected	$N^{FC} = 2$	$N^{FC} \cdot (T_2{}^{MST_0} \cdot K_2{}^{MST_0} + T_p{}^{MST_1} \cdot K_0{}^{MST_1} + 1)$
	Softmax		0

Supplementary Table 4.3 – Architecture details of OCLNN [17]. Each layer is provided with its name, main hyper-parameters and number of trainable parameters. See Sections 4.2.1 and 4.2.2 of the manuscript for the meaning of the symbols. The total number of trainable parameters was 1650, when using signals from the dataset 1, and 1874, when using signals from datasets 2 and 3. In all layers, where not specified, stride (S) and padding (P) were set to $(1,1)$ and $(0,0)$, respectively.

Block	Layer name	Hyper-parameters	Number of trainable parameters
	Input	$K_0 = 1$	0
Mixed ST (mST)	Conv2D	${K_0}^{\mathrm{mST}} = 16$, ${F_0}^{\mathrm{mST}} = (C, 9)$, ${S_0}^{\mathrm{mST}} = (1,9)$	${F_0}^{\mathrm{mST}}[0] \cdot {F_0}^{\mathrm{mST}}[1] \cdot {K_0}^{\mathrm{mST}} \cdot K_0 + {K_0}^{\mathrm{mST}}$
	ReLU		0
	Dropout	$p = 0.25$ or $p = 0.5$	0
FC	Flatten		0
	Fully-Connected	$N^{FC} = 2$	$N^{FC} \cdot ({T_0}^{\mathrm{mST}} \cdot {K_0}^{\mathrm{mST}} + 1)$
	Softmax		0

4.6.2. Details about the state-of-the-art traditional machine learning pipeline

The traditional machine learning pipeline was inspired from the re-implementation of Lawhern et al. [18] of the approach that won the BCI challenge (https://www.kaggle.com/c/inria-bci-challenge) proposed as part of the IEEE Neural Engineering Conference 2015 (code and documentation available at http://github.com/alexandrebarachant/bci-challenge-ner-2015). In particular, our pipeline consisted of 4 main steps:

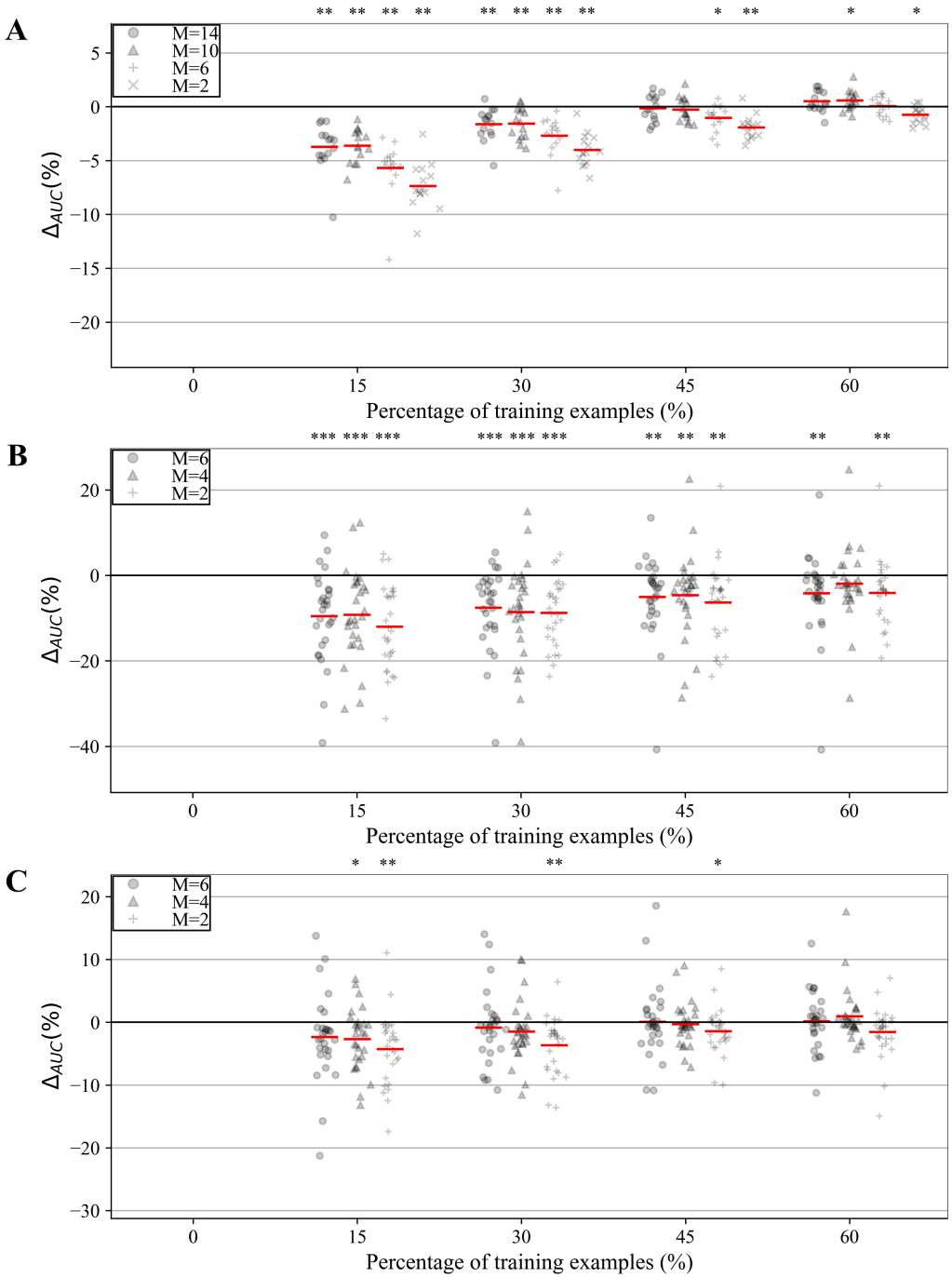
1. Learn two sets (one for the non-P300 class and one for the P300 classes) of 5 xDAWN spatial filters [40], using the ERP template concatenation method described in [41].
2. Project the covariance matrices onto the tangent space using the log-euclidean metric [42].
3. Feature normalization using a L_1 ratio of 0.5 (equal weight for L_1 and L_2 penalties).
4. Classification using Elastic Net regression using an α parameter (constant that multiplies the penalty terms) of $2 \cdot 10^{-4}$.

The pipeline was maintained the same for the three datasets, except for the number of input channels (8 for the dataset 1, 12 for datasets 2 and 3).

4.6.3. Comparison between transferring the knowledge on a small dataset and learning from scratch on the entire dataset

The main text shows the comparison between the TL-WS and the WS strategy (see Figure 4.3) using the same percentage of training examples and number of participants from whom the knowledge was transferred (M). Here, the TL-WS strategy was also compared to the WS strategy trained on 100% of training examples of the new user (corresponding to the baseline WS strategy, see Section 4.2.3-i of the manuscript). This test was performed in order to compare the results obtained while transferring the knowledge (as a function of the percentage of training examples and M) with the traditional WS strategy that used the entire training set. The statistical analysis adopted for this test is the same as the one used in Section 4.2.5-iii of the manuscript.

In Supplementary Figure 4.1, the difference between the AUC scored with the TL-WS strategy and the traditional WS strategy (i.e., using 100% of training examples, "$WS100$") is reported ($\Delta_{AUC} = AUC_{TL-WS} - AUC_{WS100}$). The performance was comparable or even significantly higher (see $M = 10$ using 60% of examples) from 45% of training examples for dataset 1, especially for higher M values. For dataset 3, the performance was comparable for each percentage in case of the highest M value ($M = 6$), from 30% in case of $M = 4$ and for 60% in case of $M = 2$. Lastly, for dataset 2, no significant difference between the two trainings strategies was obtained only when using 60% of training examples and $M = 4$ in the TL-WS strategy.



Supplementary Figure 4.1 – AUC difference between MS-EEGNet trained with the TL-WS and WS strategy using the entire training set (i.e., $\Delta_{AUC} = AUC_{TL-WS} - AUC_{WS100}$) for datasets 1-3 (Supplementary Figures 4.1A-C). The AUC difference is reported using markers and a red line denoting the mean value, as a function of the

number of participants (M) used to optimize the LOSO-M models and of the training examples of the new user (reported on the x-axis). For each percentage, a Wilcoxon signed-rank test was performed to compare TL-WS vs. WS strategy, and the statistical significance is reported (*p<0.05, **p<0.01, ***p<0.001, corrected for multiple tests) on top of each plot.

4.7. REFERENCES

[1] Sutton S, Braren M, Zubin J and John E R 1965 Evoked-Potential Correlates of Stimulus Uncertainty *Science* **150** 1187–8

[2] Polich J 2007 Updating P300: An integrative theory of P3a and P3b *Clinical Neurophysiology* **118** 2128–48

[3] Farwell L A and Donchin E 1988 Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials *Electroencephalography and Clinical Neurophysiology* **70** 510–23

[4] Nicolas-Alonso L F and Gomez-Gil J 2012 Brain Computer Interfaces, a Review *Sensors* **12** 1211–79

[5] Rezeika A, Benda M, Stawicki P, Gembler F, Saboor A and Volosyak I 2018 Brain–Computer Interface Spellers: A Review *Brain sciences*

[6] Amaral C, Mouga S, Simões M, Pereira H C, Bernardino I, Quental H, Playle R, McNamara R, Oliveira G and Castelo-Branco M 2018 A Feasibility Clinical Trial to Improve Social Attention in Autistic Spectrum Disorder (ASD) Using a Brain Computer Interface *Frontiers in Neuroscience* **12** 477

[7] Guo Y, Liu Y, Oerlemans A, Lao S, Wu S and Lew M S 2016 Deep learning for visual understanding: A review *Neurocomputing* **187** 27–48

[8] Ismail Fawaz H, Forestier G, Weber J, Idoumghar L and Muller P-A 2019 Deep learning for time series classification: a review *Data Min Knowl Disc* **33** 917–63

[9] Faust O, Hagiwara Y, Hong T J, Lih O S and Acharya U R 2018 Deep learning for healthcare applications based on physiological signals: A review *Computer Methods and Programs in Biomedicine* **161** 1–13

[10] LeCun Y, Bengio Y and Hinton G 2015 Deep learning *nature* **521** 436

[11] Lindsay G 2020 Convolutional Neural Networks as a Model of the Visual System: Past, Present, and Future *Journal of Cognitive Neuroscience* 1–15

[12] Craik A, He Y and Contreras-Vidal J L 2019 Deep learning for electroencephalogram (EEG) classification tasks: a review *J. Neural Eng.* **16** 031001

[13] Cecotti H and Graser A 2011 Convolutional Neural Networks for P300 Detection with Application to Brain-Computer Interfaces *IEEE Transactions on Pattern Analysis and Machine Intelligence* **33** 433–45

[14] Manor R and Geva A B 2015 Convolutional Neural Network for Multi-Category Rapid Serial Visual Presentation BCI *Frontiers in Computational Neuroscience* **9** 146

[15] Liu M, Wu W, Gu Z, Yu Z, Qi F and Li Y 2018 Deep learning based on Batch Normalization for P300 signal detection *Neurocomputing* **275** 288–97

[16] Farahat A, Reichert C, Sweeney-Reed C and Hinrichs H 2019 Convolutional neural networks for decoding of covert attention focus and saliency maps for EEG feature visualization *Journal of Neural Engineering*

[17] Shan H, Liu Y and Stefanov T 2018 A Simple Convolutional Neural Network for Accurate P300 Detection and Character Spelling in Brain Computer Interface *Proceedings of the 27th International Joint Conference on Artificial Intelligence* IJCAI'18 (Stockholm, Sweden: AAAI Press) pp 1604–10

[18] Lawhern V J, Solon A J, Waytowich N R, Gordon S M, Hung C P and Lance B J 2018 EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces *Journal of Neural Engineering* **15** 056013

[19] Chollet F 2016 Xception: Deep Learning with Depthwise Separable Convolutions *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 1800–7

[20] Borra D, Fantozzi S and Magosso E 2020 Convolutional Neural Network for a P300 Brain-Computer Interface to Improve Social Attention in Autistic Spectrum Disorder *XV Mediterranean Conference on Medical and Biological Engineering and Computing –*

MEDICON 2019 ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1837–43
[21] Simões M, Borra D, Santamaría-Vázquez E, GBT-UPM, Bittencourt-Villalpando M, Krzemiński D, Miladinović A, Neural_Engineering_Group, Schmid T, Zhao H, Amaral C, Direito B, Henriques J, Carvalho P and Castelo-Branco M 2020 BCIAUT-P300: A Multi-Session and Multi-Subject Benchmark Dataset on Autism for P300-Based Brain-Computer-Interfaces *Front. Neurosci.* **14** 568104
[22] Montavon G, Samek W and Müller K-R 2018 Methods for interpreting and understanding deep neural networks *Digital Signal Processing* **73** 1–15
[23] Simonyan K, Vedaldi A and Zisserman A 2014 Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps *arXiv:1312.6034 [cs]*
[24] Zhao D, Tang F, Si B and Feng X 2019 Learning joint space–time–frequency features for EEG decoding on small labeled data *Neural Networks* **114** 67–77
[25] Borra D, Fantozzi S and Magosso E 2020 EEG Motor Execution Decoding via Interpretable Sinc-Convolutional Neural Networks *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1113–22
[26] Borra D, Fantozzi S and Magosso E 2020 Interpretable and lightweight convolutional neural network for EEG decoding: Application to movement execution and imagination *Neural Networks* **129** 55–74
[27] Paszke A, Gross S, Chintala S, Chanan G, Yang E, DeVito Z, Lin Z, Desmaison A, Antiga L and Lerer A 2017 Automatic differentiation in PyTorch *NIPS-W*
[28] Vahid A, Mückschel M, Stober S, Stock A-K and Beste C 2020 Applying deep learning to single-trial EEG data provides evidence for complementary theories on action control *Commun Biol* **3** 112
[29] Ioffe S and Szegedy C 2015 Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift *Proceedings of the 32nd International Conference on Machine Learning* Proceedings of Machine Learning Research vol 37, ed F Bach and D Blei (Lille, France: PMLR) pp 448–56
[30] Clevert D-A, Unterthiner T and Hochreiter S 2015 Fast and accurate deep network learning by exponential linear units (elus) *arXiv preprint*
[31] Schirrmeister R T, Springenberg J T, Fiederer L D J, Glasstetter M, Eggensperger K, Tangermann M, Hutter F, Burgard W and Ball T 2017 Deep learning with convolutional neural networks for EEG decoding and visualization *Human brain mapping* **38** 5391–420
[32] Srivastava N, Hinton G, Krizhevsky A, Sutskever I and Salakhutdinov R 2014 Dropout: a simple way to prevent neural networks from overfitting *The Journal of Machine Learning Research* **15** 1929–58
[33] Szegedy C, Wei Liu, Yangqing Jia, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V and Rabinovich A 2015 Going deeper with convolutions *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Boston, MA, USA: IEEE) pp 1–9
[34] Supratak A, Dong H, Wu C and Guo Y 2017 DeepSleepNet: A Model for Automatic Sleep Stage Scoring Based on Raw Single-Channel EEG *IEEE Trans. Neural Syst. Rehabil. Eng.* **25** 1998–2008
[35] Blankertz B, Muller K R, Krusienski D J, Schalk G, Wolpaw J R, Schlogl A, Pfurtscheller G, Millan J D R, Schroder M and Birbaumer N 2006 The BCI Competition III: Validating Alternative Approaches to Actual BCI Problems *IEEE Trans. Neural Syst. Rehabil. Eng.* **14** 153–9
[36] Blankertz B, Muller K-R, Curio G, Vaughan T M, Schalk G, Wolpaw J R, Schlogl A, Neuper C, Pfurtscheller G, Hinterberger T, Schroder M and Birbaumer N 2004 The BCI

Competition 2003: Progress and Perspectives in Detection and Discrimination of EEG Single Trials *IEEE Trans. Biomed. Eng.* **51** 1044–51
[37] Solon A J, Lawhern V J, Touryan J, McDaniel J R, Ries A J and Gordon S M 2019 Decoding P300 Variability Using Convolutional Neural Networks *Front. Hum. Neurosci.* **13** 201
[38] Amaral C P, Simões M A, Mouga S, Andrade J and Castelo-Branco M 2017 A novel Brain Computer Interface for classification of social joint attention in autism and comparison of 3 experimental setups: A feasibility study *Journal of Neuroscience Methods* **290** 105–15
[39] Justen C and Herbert C 2018 The spatio-temporal dynamics of deviance and target detection in the passive and active auditory oddball paradigm: a sLORETA study *BMC Neurosci* **19** 25
[40] Rivet B, Souloumiac A, Attina V and Gibert G 2009 xDAWN Algorithm to Enhance Evoked Potentials: Application to Brain–Computer Interface *IEEE Trans. Biomed. Eng.* **56** 2035–43
[41] Barachant A and Congedo M 2014 A Plug&Play P300 BCI Using Information Geometry *arXiv:1409.0107 [cs, stat]*
[42] Barachant A, Bonnet S, Congedo M and Jutten C 2012 Multiclass Brain–Computer Interface Classification by Riemannian Geometry *IEEE Trans. Biomed. Eng.* **59** 920–8
[43] Glorot X and Bengio Y 2010 Understanding the difficulty of training deep feedforward neural networks *Proceedings of the thirteenth international conference on artificial intelligence and statistics* pp 249–56
[44] Kingma D P and Ba J 2017 Adam: A Method for Stochastic Optimization *arXiv:1412.6980 [cs]*
[45] Benjamini Y and Hochberg Y 1995 Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing *Journal of the Royal Statistical Society. Series B (Methodological)* **57** 289–300
[46] Medvidovic S, Titlic M and Maras Simunic M 2013 P300 Evoked Potential in Patients with Mild Cognitive Impairment *Acta Inform Med* **21** 89
[47] Vecchio F and Määttä S 2011 The Use of Auditory Event-Related Potentials in Alzheimer's Disease Diagnosis *International Journal of Alzheimer's Disease* **2011** 1–7
[48] Jeon Y-W and Polich J 2003 Meta-analysis of P300 and schizophrenia: Patients, paradigms, and practical implications *Psychophysiology* **40** 684–701
[49] Ravanelli M and Bengio Y 2018 Speaker Recognition from Raw Waveform with SincNet *2018 IEEE Spoken Language Technology Workshop (SLT)* pp 1021–8
[50] Snoek J, Larochelle H and Adams R P 2012 Practical Bayesian Optimization of Machine Learning Algorithms *Advances in Neural Information Processing Systems 25* ed F Pereira, C J C Burges, L Bottou and K Q Weinberger (Curran Associates, Inc.) pp 2951–9

SECTION II: MOTOR DECODING FROM ELECTROENCEPHALOGRAPHIC SIGNALS

CHAPTER 5: DESIGN OF AN INTERPRETABLE CNN FOR MOTOR DECODING AND ANALYSIS IN THE FREQUENCY AND SPATIAL DOMAINS

The study reported in this chapter refers to the published journal paper entitled "Interpretable and lightweight convolutional neural network for EEG decoding: Application to movement execution and imagination" D. Borra, S. Fantozzi and E. Magosso, *Neural Networks*, 2020. This study proposed for the first time the adoption of an interpretable CNN for motor decoding, obtained by increasing the interpretability of the first temporal convolutional layer (as in Chapter 3). The interpretable CNN was applied to decode motor imagery and motor execution, and the role of the main CNN components was investigated by analyzing different architecture variants. In addition, the proposed approach was compared with other CNNs and with the state-of-the-art ML algorithm. Lastly, the increased interpretability enabled the design of an intermediate explanation technique aimed to study neural signatures related to motor imagery and execution in the frequency and spatial domains.

Convolutional neural networks (CNNs) are emerging as powerful tools for EEG decoding: these techniques, by automatically learning relevant features for class discrimination, improve EEG decoding performances without relying on handcrafted features. Nevertheless, the learned features are difficult to interpret and most of the existing CNNs introduce many trainable parameters. Here, we propose a lightweight and interpretable shallow CNN (Sinc-ShallowNet), by stacking a temporal sinc-convolutional layer (designed to learn band-pass filters, each having only the two cut-off frequencies as trainable parameters), a spatial depthwise convolutional layer (reducing channel connectivity and learning spatial filters tied to each band-pass filter), and a fully-connected layer finalizing the classification. This convolutional module limits the number of trainable parameters and allows direct interpretation of the learned spectral-spatial features via simple kernel visualizations. Furthermore, we designed a post-hoc gradient-based technique to enhance interpretation by identifying the more relevant and more class-specific features. Sinc-ShallowNet was evaluated on benchmark motor-execution and motor-imagery datasets and against different design choices and training strategies. Results show that (i) Sinc-ShallowNet outperformed a traditional machine learning algorithm and other CNNs for EEG decoding; (ii) The learned spectral-spatial features matched well-known EEG motor-related activity; (iii) The proposed architecture performed better with a larger number of temporal kernels still maintaining a good compromise between accuracy and parsimony, and with a trialwise rather than a cropped training strategy. In perspective, the proposed approach, with its interpretative capacity, can be exploited to investigate cognitive/motor aspects whose EEG correlates are yet scarcely known, potentially characterizing their relevant features.

5.1. INTRODUCTION

Approaches based on machine learning algorithms provide powerful tools to analyze and decode brain activity from electroencephalographic (EEG) data, both in research and application areas. In particular, machine learning techniques have been exploited in many EEG-based Brain-Computer Interfaces (BCIs). In these systems, a feature extraction stage [1] extracts the meaningful characteristics of the pre-processed [2] EEG signals and a downstream classification stage [3] makes a decision based on the extracted characteristics, to provide the appropriate feedback to the user [4]. One popular and performing feature extraction algorithm is the filter bank common spatial pattern (FBCSP) (Ang, Chin, Zhang, & Guan, 2008) that applies a bank of bandpass filters (selected a priori) and extracts features for each frequency band based on the spatial filtering method. FBCSP has been widely used as EEG feature extraction method and won several competitions, such as BCI competition IV datasets 2a and 2b [5] related to EEG decoding of imagined movements.

However, the traditional machine learning pipeline described above performs feature extraction and classification in separate steps. Furthermore, it strongly relies on a priori knowledge in the design of the feature extraction stage (e.g., the filters' cut-off frequencies in the FBCSP) and prevents that other potentially relevant (but unknown) features are extracted and used for decoding. For this reason, this approach may also have negative impact on decoding accuracy. Recently, machine learning innovations, proposed in the computer vision field and represented by convolutional neural networks (CNNs), have been transposed to EEG decoding tasks [6], mitigating the need for manual feature extraction. CNNs automatically learn features in a hierarchical structure from the input data in an end-to-end fashion, i.e., without separating the feature extraction, selection and classification steps. Thus, in the field of EEG decoding, CNNs can be trained by feeding EEG signals as input to the neural network, obtaining as output the corresponding predicted label. Accordingly, CNNs do not need any a priori knowledge about the meaningful characteristics of the signals for the specific decoding task and have the potentiality to discover the relevant features (even so-far unknown) by using all input information.

An efficient way to provide EEG signals as input to CNNs is to design a 2D input representation with the electrodes along one dimension and time steps along the other [7–18], preserving the original EEG representation i.e., non-transformed representation. Other input representations, e.g., transformed representations such as time-frequency decomposition [19–21], generally increase data dimensionality requiring more training data and/or regularization to learn meaningful features. CNNs with a non-transformed representation are typically designed by stacking individual temporal and spatial convolutional layers or a single spatio-temporal convolutional layer, and eventually deeper convolutional layers that learn patterns on the spatially filtered activations. CNNs based on these architectures have been successfully applied to several EEG decoding tasks, such as P300 detection tasks [7,9–11,13,15], motor imagery and execution decoding tasks [14,11,16,18,8], anomaly detection tasks [12], emotion classification [17], and they have been generally proved to outperform traditional machine learning approaches. Despite these effective applications of CNNs in EEG decoding, there are still a number of critical issues that require further investigation. Indeed, CNNs introduce a large number of trainable parameters requiring large training datasets to obtain a good fit, have

a longer training time compared to simpler models, introduce many hyper-parameters (e.g., number of kernels, kernel sizes, number of layers, type of activation functions, etc.), and the automatically learned features are difficult to be interpreted. In particular, techniques that increase the interpretability of the learned features are receiving growing interest as key ingredients to achieve more robust validation when using CNNs [22]. In the field of CNN-based EEG decoding, increasing the interpretability may be particularly relevant for neuroscientists as to the following aspects: (i) check the correct learning by verifying that the models do not rely on artefactual sources but on neurophysiological features; (ii) enable the understanding of which EEG features better discriminate the investigated classes; (iii) potentially characterize new features exploited by the network for the classification, and thus increase the insight into the neural correlates underlying the classified behaviors.

Several efforts have been made to increase CNN interpretability via post-hoc interpretation techniques (i.e., techniques that analyze the trained model). These techniques include temporal and spatial kernel visualizations [9,11], kernel ablation tests (i.e., selective removal of single kernels) [11], saliency maps (i.e., maps showing the gradient of CNN prediction with respect to its input example) [10], gradient-weighted class activation mapping [23], correlation maps between input features and outputs of given layers [14]. Some of these works face the interpretability issue together with other key issues previously cited, such as model complexity (in terms of number of layers and numbers of trainable parameters) and the size of the training dataset. Schirrmeister et al. [14] tested both a deeper CNN (DeepConvNet, with 5 convolutional layers and one fully-connected layer) and a shallower CNN (ShallowConvNet, with 2 convolutional layers and one fully-connected layer) for decoding movement execution and motor imagery, analyzed the effect of increasing the amount of training examples (via cropped training), and used correlation maps to interpret the CNN learned features. Lawhern et al. [11] designed a shallow and lightweight CNN (EEGNet, with 3 convolutional layers and one fully-connected layer) by introducing depthwise and separable convolutions that reduced the number of parameters to fit, tested a range of EEG decoding tasks with various training sizes, and interpreted the learned features via kernel visualization and ablation.

Besides post-hoc techniques, network interpretability may be increased by introducing directly interpretable layers within the network architecture; importantly, these layers may intrinsically reduce the number of trainable parameters too, promoting more interpretable and, at the same time, lightweight CNNs. Very recently, few studies have explored this approach in CNNs for EEG decoding. Zhao et al. [18] introduced a time-frequency convolutional layer in an architecture inspired by ShallowConvNet [14] to learn time-frequency filters designed by real-valued Morlet wavelets. In a previous preliminary work [8], for the first time we used a temporal sinc-convolutional layer [24] for EEG decoding, included in an architecture based on DeepConvNet [14], to learn temporal filters defined by parametrized sinc-functions that implement band-pass filters. Instead of learning all the kernel values as in a traditional convolutional layer, both in the wavelet- and sinc-convolutional layer only 2 parameters for each kernel need to be learned and they are directly interpretable: the bandwidth of the Gaussian and the wavelet central frequency in one case [18], and the two cutoff frequencies of the band-pass filters in the other case [8]. While this approach appears promising, its use in EEG decoding is still limited and the so-far proposed CNNs [8,18] have some limitations. Indeed, except for a single directly interpretable convolutional layer, the rest of these CNNs

uses traditional less interpretable convolutional layers. This aspect, not only may hinder the overall interpretability of the learned features, but also requires a large number of trainable parameters leading to models more prone to overfitting and this is especially true in case of the deep CNN we previously proposed [8]. Furthermore, each of these CNNs has been tested only on a single decoding task (movement imagination [18], and movement execution [8]), and the ability of each network to generalize across motor paradigms has not been verified.

The purpose of this work is to contribute to the recent developments of CNN-based EEG decoding by designing and analyzing a novel CNN that includes interpretable and optimized layers, able to increase the overall interpretability of the network, reduce the number of trainable parameters and, at the same time, ensure good performances compared to existing state-of-the art (SOA) algorithms. The CNN proposed here is a lightweight shallow CNN, named Sinc-ShallowNet, obtained by stacking two convolutional layers that extract spectral and spatial EEG features respectively, followed by a fully-connected layer finalizing the classification. The two convolutional layers are specifically devised to increase interpretability and decrease the number of trainable parameters and consist of a temporal sinc-convolutional layer and a spatial depthwise convolutional layer. The spatial depthwise convolutional layer ties spatial filters to each particular band-pass filter learned by the temporal sinc-convolutional layer, enabling the learning of spatial features related to specific frequency ranges. The proposed architecture was applied to decode sensorimotor rhythms both during motor execution (ME) and motor imagery (MI) using public benchmark datasets. Moreover, an extensive analysis of Sinc-ShallowNet was performed including the following aspects:

i. Comparison of the decoding performance of Sinc-ShallowNet with SOA decoding algorithms, including one traditional machine learning pipeline based on FBCSP coupled with regularized Linear Discriminant Analysis (rLDA) and other three CNNs (ShallowConvNet and DeepConvNet [14], EEGNet [11]).
ii. Assessment of some design choices on Sinc-ShallowNet performance in a post-hoc hyper-parameter evaluation procedure inspired by Schirrmeister et al. [14]. The evaluated design choices concern: the number of the temporal band-pass filters, the number of spatial filters for each temporal filter, the introduction of an optional recombination of the spatial activations, and the size of activation aggregation (average pooling) before the fully-connected layer.
iii. Evaluation of the effect of increasing the training data size via cropped training compared to trialwise training. Indeed, the effect of cropped training on different CNN architectures is still unclear. Schirrmeister et al. [14] found that cropped training significantly increased the performance of deep architectures (DeepConvNet), while no significant effect was obtained with shallow architectures (ShallowConvNet). Despite this, other shallow architectures [18] were trained with a cropped strategy. Therefore, we evaluated the effect of the training strategy on the performance of Sinc-ShallowNet and of the re-implemented SOA CNNs.
iv. Feature interpretation. Since the trainable parameters of the temporal sinc-convolutional layer are the cutoff frequencies of the learned band-pass filters, the learned spectral features can be directly visualized and interpreted once the training ends. Furthermore, inspired from the saliency maps [25], we designed a post-hoc interpretation technique named “temporal sensitivity analysis” (as it acts on the kernels of the temporal sinc-

convolutional layer). This technique enables the identification of the more relevant and more class-specific band-pass filters and the spatial features (as learned in the depthwise convolutional layer) related to these band-pass filters can be visualized.

5.2. MATERIALS AND METHODS

This section is devoted to the description of the proposed CNN for EEG motor decoding. At first, we define the problem of EEG decoding into the framework of supervised classification learning via CNNs and provide notations useful for the following description. Subsequently, we illustrate the benchmark datasets used to train and test the CNNs (the proposed one and the SOA CNNs), the architecture of the proposed CNN, the training procedure, and finally the post-hoc interpretation technique. The CNNs were developed in PyTorch [26] and trained from scratch using a workstation equipped with an AMD Threadripper 1900X, NVIDIA TITAN V and 32 GB of RAM.

5.2.1. Problem definition and notations

Let us assume to have an EEG dataset collected from each subject. Each dataset consists of separated trials (e.g., obtained by epoching the original continuous EEG recording), with each trial belonging to one of several classes (let's say N_c classes). By indicating with $M^{(s)}$ the total number of trials for s-th subject, the corresponding dataset can be denoted by $D^{(s)} = \left\{\left(X_0^{(s)}, y_0^{(s)}\right), \left(X_2^{(s)}, y_2^{(s)}\right), \dots, \left(X_{M^{(s)}-1}^{(s)}, y_{M^{(s)}-1}^{(s)}\right)\right\}$. $X_i^{(s)} \in \mathbb{R}^{C \times T}$ contains the pre-processed EEG signals of the i-th trial ($0 \leq i \leq M^{(s)} - 1$), collected at C electrodes and T time samples; $y_i^{(s)}$ is the class label of the i-th trial and assumes one value among the N_c possible values, i.e., $\forall\, i, y_i^{(s)} \in L = \{l_0, l_2, \dots, l_{N_c-1}\}$. The two public EEG datasets used here were EEG signals collected while the subjects executed (High-Gamma dataset, see Section *Motor execution: High-Gamma dataset*) or imagined (BCI-IV2a dataset, see Section *Motor imagery: BCI-IV2a dataset*) movements of different body parts. Thus, the classes discriminate among the specific body part moved (or imagined to be moved) during each trial (e.g., l_0 = "Right Hand", l_1 = "Left Hand" etc.).

The problem at hand is to train a classifier f so that it learns, from a training set of EEG trials, to assign the correct label to previously unseen EEG trials. Specifically, the parametric classifier is $f\left(X_i^{(s)}; \theta^{(s)}\right) : \mathbb{R}^{C \times T} \rightarrow L$, parametrized by parameters $\theta^{(s)}$, which assigns a label $y_i^{(s)}$ to the trial $X_i^{(s)}$, i.e., $y_i^{(s)} = f\left(X_i^{(s)}; \theta^{(s)}\right)$. The classifier $f\left(X_i^{(s)}; \theta^{(s)}\right)$ can be formally interpreted as the composition of two functions: (i) a first function ϕ that extracts a (vector-valued) feature representation $\phi\left(X_i^{(s)}; \theta_\phi^{(s)}\right): \mathbb{R}^{C \times T} \rightarrow \mathbb{R}^{N_\phi}$ (N_ϕ denoting the number of extracted features) having parameters $\theta_\phi^{(s)}$; (ii) a second function $g\left(\phi^{(s)}; \theta_g^{(s)}\right): \mathbb{R}^{N_\phi} \rightarrow L$ with parameters $\theta_g^{(s)}$ that exploits the extracted features to finalize the classification, that is $f\left(X_i^{(s)}; \theta^{(s)}\right) = g\left(\phi\left(X_i^{(s)}; \theta_\phi^{(s)}\right); \theta_g^{(s)}\right)$. When the decoder f is implemented by a CNN, the two stages (feature extraction and final classification) are learned jointly with all parameters $\theta^{(s)}$ optimized simultaneously. By keeping superscript s in the classifier parameters, we emphasize that the parameters are optimized separately per subject, as here a within-subject training procedure (see Section 5.2.4) was adopted. The overall set of trials $D^{(s)}$ of each subject is divided into a training set, used to optimize the parameters $\theta^{(s)}$ for the specific subject, and a test set used to evaluate the performance of the learned subject-specific decoder.

Of course, besides the trainable parameters $\theta^{(s)}$, the network hyper-parameters (i.e., parameters that define the functional form of decoder f not adapted by the learning itself, such as the number of layers, number and size of convolutional kernels, type of activation function, etc.) may affect the decoding accuracy.

In the following, we assume that the generic trial $X_i^{(s)} \in \mathbb{R}^{C \times T}$ is given in input to the CNNs as a 2D matrix of shape (C, T), having the time steps along the width and the electrodes along the height.

5.2.2. Datasets

The datasets used in this study are two common benchmark MI- and ME-EEG datasets for sensorimotor rhythm decoding. It is known that the α, β and γ bands are associated with movement-related spectral power modulations and thus provide class-discriminative information [27–34]. In the following, these datasets are described together with the light pre-processing applied to obtain the trials $X_i^{(s)}$ used to train and test the CNNs.

Motor execution: High-Gamma dataset

High-Gamma dataset is a 128-channel ME-EEG dataset acquired from 14 healthy subjects (age 27.2±3.6, 6 female, 2 left-handed) by Schirrmeister et al. [14] and made freely available (https://web.gin.g-node.org/robintibor/high-gamma-dataset). Each subject performed roughly 1000 (963.1±150.9 mean ± standard deviation (std) across participants) four-second trials of movement execution (three different movements) and of rest. The three movements were repetitive right- and left-hand sequential finger tapping, and repetitive toes clenching. Therefore, the decoding problem is a 4-way classification task. This dataset is well-suited for extracting information from the high γ band (> 50Hz) since it was acquired in a laboratory optimized for the recording of high-frequency movement-related EEG components [14]. High-frequency components, approx. between 60-90 Hz were found modulated during movement execution and may contain useful movement-related information [27, 28, 34]. Therefore, this dataset could be exploited to investigate neural correlates in γ band in addition to the well-known motor modulations in α and β bands [29-33].

EEG signals were downsampled from 500 to 250 Hz, the same sample frequency as the other analyzed dataset (see Section *Motor imagery: BCI-IV2a dataset*), so that the CNN hyper-parameters related to the temporal dimension (i.e., temporal kernel and pooling sizes) were kept the same. The 44 signals relative to the electrodes covering the motor cortex were selected (Figure 5.1a) as done in [14] and a high-pass 3rd order Butterworth filter was applied with a cutoff frequency of 4 Hz. Then, each electrode signal was standardized by applying an exponential moving average window with a decay factor of 0.999 as done in [14]. Each signal was epoched between -0.5 and 4.0 s relative to the movement onset, so that each trial contains EEG values at $C = 44$ electrodes and at $T = 1125$ time samples organized in a single input feature map ($K = 1$, denoting with K the number of the input feature maps). Finally, the resulting trials were cleaned from high-amplitude artefacts by removing those with at least one electrode signal with absolute value $> 800\mu V$. Based on the previous description, the CNN input (corresponding to a single trial) had shape $(K, C, T) = (1,44,1125)$ in this case.

For the sake of reproducibility of the results, the trial set $D^{(s)}$ of the s-th subject was split as in the original paper [14] for training and testing: for each subject, 160 trials (40 for each class) were used as test set and the remaining as training set. In addition, the training set was further split into a validation set (20% of the training set) in order to perform early stopping during the first step of the optimization process (see Section 5.2.4).

Motor imagery: BCI-IV2a dataset

BCI-IV2a dataset is a 22-channel MI-EEG dataset collected for the BCI Competition IV [35]. This set comprises four classes of imagined movements of left and right hands, feet and tongue, acquired from 9 participants and made freely available (http://www.bbci.de/competition/iv/). Therefore, the decoding problem is a 4-way classification task. The organizers of the challenge provided the dataset sampled at 250 Hz and band-pass filtered between 0.5 and 100 Hz. All 22 signals were used, and the montage is shown in Figure 5.1b.

The EEG signals were band-pass filtered between 4 and 38 Hz with a 3rd order Butterworth filter and each electrode signal was standardized by applying an exponential moving average window with a decay factor of 0.999 [14]. Then, the signals were epoched between 0.5 and 2.5 s relative to the movement onset of movement imagination, as done in previous studies [11,20,36]. In this case, the CNN input (i.e., the single trial) had shape $(1,22,500)$.

Here we used the same training set (288 trials per subject, balanced between the classes) and test set (288 trials per subject, balanced between the classes) provided by the organizers of the competition. The training set was further split into a validation set (20% of the training set) in order to perform early stopping during the first step of the optimization process (see Section 5.2.4).

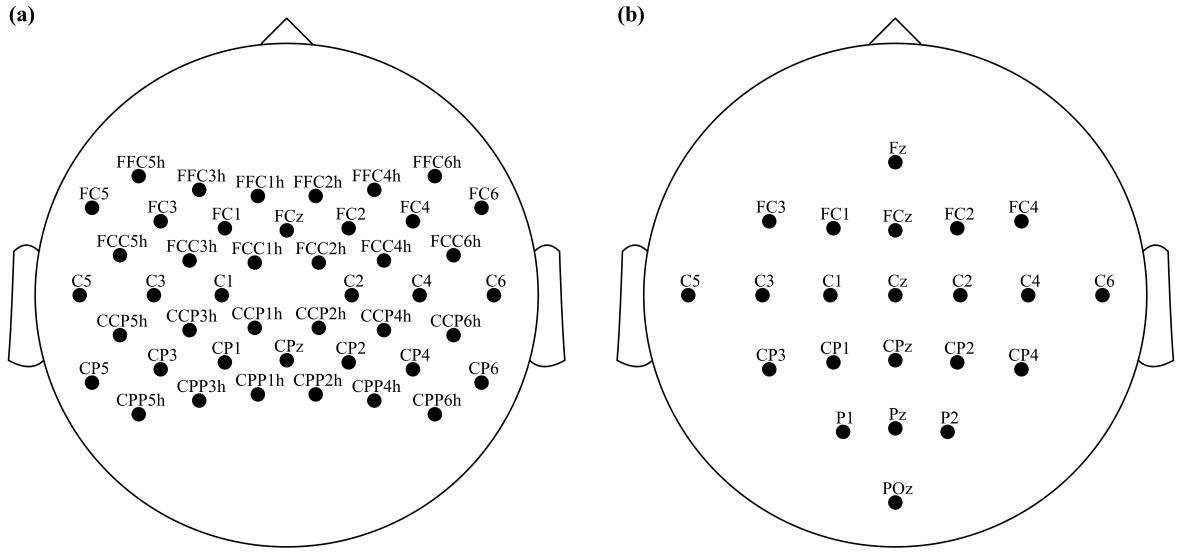


Figure 5.1 – Electrode locations for the two examined datasets. (a) ME-EEG dataset. (b) MI-EEG dataset.

5.2.3. Sinc-ShallowNet

The proposed architecture is designed with three fundamental blocks, each of them composed by a few layers. The blocks of the proposed architecture and their fundamental layers

are shown in Figure 5.2; a detailed description of the architecture (including the name, output shape and number of trainable parameters of each layer) is reported in Table 5.1. Block 1 has the function to extract spectral and spatial features from the input data, via temporal and spatial convolutional layers, respectively. The performed convolutions are designed to reduce the number of trainable parameters while increasing their interpretability. As to the temporal convolution, this is achieved via a sinc-convolutional layer (see Section *Sinc-convolutional layer*), while for the spatial convolution, this is achieved via a depthwise convolutional layer [37]. Block 2 is devoted to perform a temporal aggregation (via a pooling layer) of the first block feature maps. Block 3 is designed to finalize the classification including a single fully-connected layer. The term "sinc" of Sinc-ShallowNet is related to the inclusion of the temporal sinc-convolutional layer within the first block; the term "shallownet" refers to the relative low number of trainable layers.

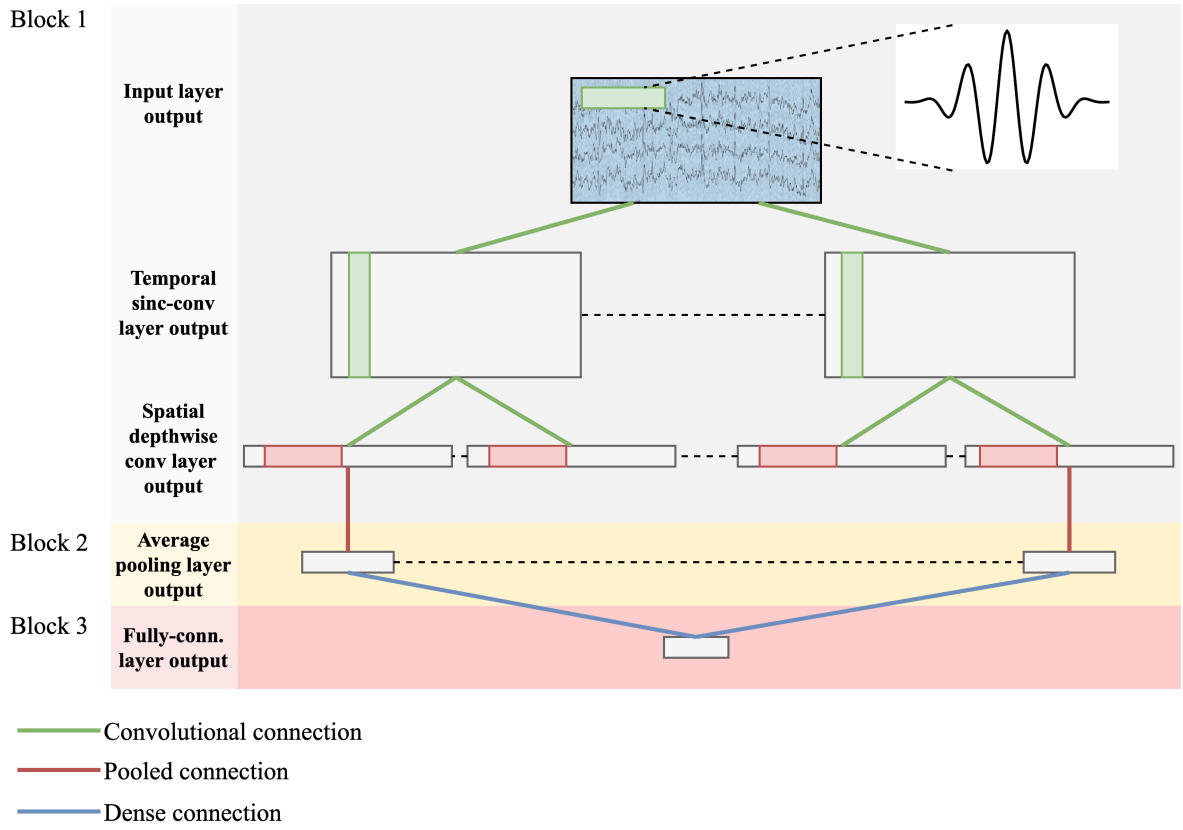


Figure 5.2 – Architecture of Sinc-ShallowNet. For simplicity, the figure shows only the more significant layers within each of the three blocks (see also Sections *Block 1: Spectral and spatial feature extraction*, *Block 2: Aggregation*, *Block 3: Classification* and Table 5.1).

Table 5.1 – Architecture details of Sinc-ShallowNet. The architecture corresponding the hyper-parameters reported here is denoted as "basal" Sinc-ShallowNet (variants of this basal architecture are also tested, see Table 5.2). Each layer is provided with its name, main hyper-parameters, output shape and number of trainable parameters and adopted activation function. C and T represent the number of electrodes and time samples of the network input, respectively. N_c is the number of the classes. See Section 5.2.3 for the meaning of the other symbols. The output shapes of the layers within the first and second blocks are described by tuples of three integers (in brackets) denoting the number of feature maps (CNN channel dimension) and the number of spatial and temporal samples within each map, respectively. The input layer provides an output of shape $(1, C, T)$ since it is assumed to just replicate the original input matrix with shape (C, T), providing a single feature map as output (coincident with its input). The output shapes in the third block are 1D, thus described by a single number. *Kernel maximum norm constraint was used, enforcing an absolute upper bound on the magnitude of the weights.

Block	Layer name	Hyper-parameters	Output shape	Number of parameters	Activation
1	Input		$(1, C, T)$	0	
	Sinc-Conv2D	$K_1 = 32$ $F_1 = (1,65)$ $S_1 = (1,1)$ $P_1 = (0,0)$	(K_1, C, T_1)	$2 \cdot K_1$	Linear
	BatchNorm2D	$m = 0.99$	(K_1, C, T_1)	$2 \cdot K_1$	
	DW-Conv2D*	$K_2 = K_1 \cdot D_2$ $F_2 = (C, 1)$ $D_2 = 2$ $S_2 = (1,1)$ $P_2 = (0,0)$	$(K_2, 1, T_1)$	$F_2[0] \cdot K_2$	Linear
2	BatchNorm2D	$m = 0.99$	$(K_2, 1, T_1)$	$2 \cdot K_2$	
	Activation	$\alpha = 1$	$(K_2, 1, T_1)$	0	ELU
	AvgPool2D	$F_p = (1,109)$ $S_p = (1,23)$	$(K_2, 1, T_p)$	0	
	Dropout	$p = 0.5$	$(K_2, 1, T_p)$	0	
3	Flatten		$(K_2 \cdot T_p)$	0	
	Fully-Connected*	$N_c = 4$	(N_c)	$N_c \cdot T_p \cdot K_2 + N_c$	
	Activation		(N_c)	0	Softmax

In the first two blocks, the output of each layer can be interpreted as a collection of spatio-temporal feature maps. Thus, its shape can be described by a tuple of 3 integers, with the first integer indicating the number of feature maps provided by the layer, the second and third integers the number of spatial and temporal samples within each map, respectively. Each convolutional layer in these blocks is characterized by the number of convolutional kernels (K), kernel size (F), stride size (S), and padding size (P). In addition, depthwise convolution introduces also a depth multiplier (D), that specifies the number of kernels to learn for each input feature map. Since Sinc-ShallowNet has two convolutional layers, the previous symbols are provided with subscript ("1", "2"). The pooling layer in block 2 is described by the pool size (F_p) and pool stride (S_p). Since the adopted convolutions and pooling are 2D, the hyper-parameters F_i, S_i, P_i $(i = 1, 2)$, F_p, S_p are tuples of two elements: the first element refers to the spatial dimension, while the second to the temporal dimension.

Block 1 and block 2 stacked together can be seen as implementing the function $\phi\left(X_i^{(s)};\theta_\phi^{(s)}\right)\colon \mathbb{R}^{C\times T}\to\mathbb{R}^{N_\phi}$ (described in Section 5.2.1), where N_ϕ is the overall number of units provided as output by block 2. Block 3 receives this flattened feature map and finalizes the classification, implementing a dense softmax classification. Thus, this block realizes the function $g\left(\phi^{(s)};\theta_g^{(s)}\right)\colon \mathbb{R}^{N_\phi}\to L$ (described in Section 5.2.1). Of course, all parameters of the three blocks are optimized simultaneously during the training, without any separation between the feature extraction and classification stages.

In the following, we will first describe the mathematical aspects of the temporal sinc-convolutional layer and the motivation for its inclusion. Then, the structure and function of each block will be detailed.

Sinc-convolutional layer

Recently, Ravanelli and Bengio (2018) designed a CNN for speaker recognition (SincNet) including a "sinc-convolutional layer", that forces each kernel to describe a band-pass filter. In a traditional convolutional layer, each value of a kernel is learned during the optimization. In a sinc-convolutional layer, each value of a kernel is defined by a parametrized function, forcing the kernel description to belong to a specific subset of temporal filters (here only band-pass filters) and at the same time reducing the number of trainable parameters. This implementation promotes the learning of more meaningful and well-defined temporal filters.

Considering the i-th electrode signal x_i (here, for simplicity the superscript s referring to a specific subject is omitted), the 1D convolution between this signal and the j-th kernel k_j is (Equation 5.1):

$$o_{i,j}[n] = x_i[n] * k_j[n] = \sum_{l=0}^{F-1} x_i[n-l]\cdot k_j[l], \tag{5.1}$$

where $i \in [0, C-1]$ with C representing the number of EEG electrodes, $j \in [0, K-1]$ with K representing the number of temporal kernels, and F is the kernel size. Since, for brevity, we are describing a 1D convolution, here F is 1D (i.e., F represents the length of the filter along the temporal dimension). For instance, let's say $F = 65$ for capturing frequencies at ~ 4 Hz and above in case of data at 250 Hz sampling rate [11].

The kernel values of a sinc-convolutional layer can be obtained by evaluating the parametrized function ${k_j}'[n;\theta_j]$ with a specific set of trainable parameters θ_j defining the j-th band-pass filter. To describe band-pass filters in the frequency domain, the amplitude ${K_j}'$ can be expressed as (Equation 5.2):

$${K_j}'[f; f_{1,j}, f_{2,j}] = rect\left(\frac{f}{2f_{2,j}}\right) - rect\left(\frac{f}{2f_{1,j}}\right), \tag{5.2}$$

where $\theta_j = \{f_{1,j}, f_{2,j}\}$ is the set of the trainable parameters of the j-th kernel. This set includes only the inferior ($f_{1,j}$) and the superior ($f_{2,j}$) cutoff frequencies of the j-th band-pass filter, reducing the number of trainable parameters of the temporal convolutional layer from $F = 65$ to 2 for each temporal kernel. In the temporal domain, ${k_j}'$ can be expressed as (Equation 5.3):

$${k_j}'[n; f_{1,j}, f_{2,j}] = 2f_{2,j}sinc\left(2\pi f_{2,j}n\right) - 2f_{1,j}sinc\left(2\pi f_{1,j}n\right). \tag{5.3}$$

To alleviate the effects of the inevitable truncation of $k_j{}'$ on the characteristics of the filters (e.g., passband ripples, reduced stopband attenuation), the function is multiplied by a Hamming window (Equation 5.4) [24]:

$$\begin{cases} k_{w,j}{}'[n; f_{1,j}, f_{2,j}] = k_j{}'[n; f_{1,j}, f_{2,j}] \cdot w[n] \\ \quad w[n] = 0.54 - 0.46 \cos\left(\frac{2\pi n}{F-1}\right) \end{cases} . \tag{5.4}$$

The so-defined convolutional layer can be integrated into a traditional CNN to learn band-pass filters in the first layer, with only the two cutoff frequencies as trainable parameters. In this study, these frequencies were randomly initialized from a uniform distribution in the frequency range of interest: (4,125] Hz and (4,38] Hz for ME- and MI-EEG signals, respectively. During the optimization, these frequencies were updated in the range of interest by keeping $f_{2,j} > f_{1,j}$.

Block 1: Spectral and spatial feature extraction

The first block (see Figure 5.2 and Table 5.1) performed a separate spectral and spatial feature learning. The first layer of this block was a 2D temporal sinc-convolutional layer that learned $K_1 = 32$ band-pass temporal filters with a low number of learnable parameters. The filter size F_1 was set to $(1,65)$ to extract information at 4 Hz and above, since the CNN input data were high-pass filtered at 4 Hz in the pre-processing stage (see Section 5.2.2). Following this layer, batch normalization (see Section 5.2.5) [38] was introduced along the feature map dimension. Then, a 2D spatial depthwise convolutional layer was introduced to learn $D_2 = 2$ spatial filters of size $(C, 1)$ for each temporal feature map, with a total number of $K_2 = K_1 \cdot D_2$ spatial filters. The depthwise convolution is not fully-connected with the previous temporal feature maps (see Figure 5.2), reducing the number of trainable parameters. Moreover, it allows a straightforward extraction of the spatial distribution of each band-pass filter, making the interpretation of the learned CNN features easier. In this layer, kernel maximum norm constraint was used.

Block 2: Aggregation

The second block (see Figure 5.2 and Table 5.1) was designed to perform a temporal aggregation of the first block output. First, batch normalization (see Section 5.2.5) [38] along the feature map dimension was applied to the neurons of the spatial depthwise convolutional layer, followed by a non-linear activation function. In this study, Exponential Linear Units (ELUs) were adopted with activation function $f(x) = x, x > 0$ and $f(x) = \alpha \cdot (exp(x) - 1), x \leq 0$, as this non-linearity allows faster and more noise-robust learning than other non-linearities [39]. Furthermore, Schirrmeister et al. [14] reported better performance for shallow and deep CNNs applied to EEG motor decoding when using ELUs compared to other activation functions. The α parameter is the ELU hyper-parameter that controls the saturation value for negative inputs and $\alpha = 1$ was set for the proposed architecture. Then, an average pooling layer was introduced to reduce the number of trainable parameters in the transition from the block 2 and the subsequent fully-connected layer in block 3, i.e., the convolutional-to-dense connections. A pool size of $F_p = (1, 109)$ and pool stride of $S_p = (1, 23)$ were used. These

hyper-parameters allow the extraction of averaged spatial activations of ~ 500 ms with a stride of ~ 100 ms. Lastly, a dropout layer [40] was introduced (see Section 5.2.5).

Block 3: Classification

After the second block, a flatten layer was introduced to unroll the second block output values, resulting in a 1D array of features extracted by the previous layers. These values are densely connected with a single fully-connected layer containing $N_c = 4$ neurons.

Accordingly, the entire CNN maps the input data of the i-th trial $X_i^{(s)}$ to one real number per class, i.e., $h\left(X_i^{(s)}; \theta^{(s)}\right)$: $\mathbb{R}^{C \times T} \longrightarrow \mathbb{R}^{N_c}$. These N_c outputs are transformed via a softmax activation function to obtain the conditional probabilities of the labels $l_k \; \forall k \in L = \{l_0, l_1, \dots, l_{N_c-1}\}$: $p\left(l_k \middle| X_i^{(s)}, \theta^{(s)}\right) = \frac{\exp h_k(X_i^{(s)}; \theta^{(s)})}{\sum_{j=0}^{N_c-1} h_j(X_i^{(s)}; \theta^{(s)})}$. Since the training strategy adopted was a within-subject training (see Section 5.2.4), the softmax provides subject-specific conditional distribution over the N_c classes for each example. The final classification is performed by assigning the label with the maximum conditional probability to the trial $X_i^{(s)}$, i.e., $y_i^{(s)} = f\left(X_i^{(s)}; \theta^{(s)}\right) = arg \; \underset{l_k}{max} \, p\left(l_k \middle| X_i^{(s)}, \theta^{(s)}\right)$.

Based on the number of trainable parameters within each layer (see Table 5.1), Sinc-ShallowNet introduced a total number of trainable parameters of 13828 and 5508, for ME- and MI-EEG signals, respectively.

Design choices

In the following, the Sinc-ShallowNet described as in the previous sections (with the corresponding hyper-parameters, see Table 5.1) will be denoted as the "basal" Sinc-ShallowNet. In order to evaluate the influence of specific hyper-parameters of interest on the performance metric, a post-hoc hyper-parameter evaluation was performed by testing some variants compared to the basal architecture. The investigated hyper-parameters were: (i) the number of the temporal filters in block 1 (K_1); (ii) the number of the spatial filters per temporal filter in block 1 (D_2); (iii) the pooling size F_p and stride S_p of the average pooling in block 2; (iv) the recombination of the spatial activations. In the condition (iv), a pointwise convolution was included as the first layer in block 2 (fed by the outputs of the spatial depthwise convolution), followed by the other layers of block 2 (i.e., batch normalization, non-linear activation, average pooling, dropout).

These alternative design choices were evaluated through an extensive experimentation as described and motivated in Table 5.2.

Table 5.2 – Investigated design choices.

Architectural aspect	Basal	Variants	Motivation
Number of temporal filters K_1 of block 1	$K_1 = 32$	$K_1 = 8$ $K_1 = 16$	We wanted to test if lowering the number of the temporal kernels worsened the performance, i.e., to check if all the 32 temporal filters were needed or some of them were redundant. Furthermore, since

			there is a consistent variability in the number of temporal kernels within CNNs for EEG decoding (e.g., 8 in EEGNet, 25 in DeepConvNet, 40 in ShallowConvNet), this test on Sinc-ShallowNet may gain insights about the effect of this hyper-parameter on the decoding performance. Of course, a larger number of the band-pass filters implied a larger number of trainable parameters but this effect was limited since the sinc-convolutional layer learns only 2 parameters for each temporal filter.
Number of spatial filters per temporal filter D_2 of block 1	$D_2 = 2$	$D_2 = 4$	We wanted to test if increasing the number of the spatial filters for each band-pass filter increased the performance. We expected that a higher D_2 was more beneficial for those applications in which the band-pass kernels were more dispersed across a large frequency range, i.e., in case the signals contained more frequency components, such as the investigated ME-EEG signals. In case of less dispersed band-pass filters, there is high probability that neighbor band-pass kernels are learned; the neighbor band-pass kernels can compensate for the reduction in D_2 as they may be tied with different spatial filters learned during training, actually providing an augmented set of spatial filters for a given band-pass filtering. The drawback of an increase of D_2 was an increased number of trainable parameters.
Pooling size F_p and stride S_p of block 2	$F_p = (1{,}109)$ $S_p = (1{,}23)$	$F_p = (1{,}71)$ $S_p = (1{,}15)$	We wanted to evaluate the impact of a shorter average pooling on the performance. The modified values of these hyper-parameters corresponded to the extraction of averaged spatial activations of 325 ms with a stride of 70 ms (similarly as done in [14,18]). This variant resulted in an increased number of trainable parameters due to a convolutional-to-dense transition involving more units.
Recombination of the spatial activations via an additional pointwise convolution in block 2	No recomb.	Recomb.	We wanted to evaluate the impact of the recombination of the spatial activations on the performance. A pointwise convolutional layer was introduced immediately after the spatial depthwise convolutional layer, in order to recombine the learned spatial activations across the feature map dimension. The hyper-parameters of this layer were $K_3 = K_2 = K_1 \cdot D_2$, $F_3 = (1{,}1)$, $S_3 = (1{,}1), P_3 = (0{,}0)$. The combination of a depthwise and a pointwise convolution is called separable convolution [37]. The introduction of pointwise convolution increase the number of trainable parameters by $(K_2)^2$ and the resulting architecture may need a large training set. Thus, this modification could be more beneficial in case of the investigated ME-EEG signals.

From the specifications reported in Table 5.2, five variants of Sinc-ShallowNet were designed by changing one specific hyper-parameter at a time while keeping all other hyper-parameters fixed, as previously done in Schirrmeister et al. [14] and Farahat et al. [10], and were trained as specified in Section *Trialwise training strategy*.

5.2.4. Training

Trialwise training strategy

For each subject, the entire trial was used as input and the corresponding trial label as target to optimize one CNN per subject (within-subject training). Weights were randomly initialized adopting a Xavier uniform initialization scheme [41] and biases were initialized to zero. The cutoff frequencies of the temporal sinc-convolutional layer were initialized as described previously (see Section *Sinc-convolutional layer*). The trainable parameters $\theta^{(s)}$ were optimized such that the parametric classifier assigned high probabilities to the correct labels by minimizing the sum of the per-example losses computed on the N training examples, converging to an optimal trainable parameter set $\theta^{(s)*}$ (Equation 5.5):

$$\theta^{(s)*} = \arg\min_{\theta^{(s)}} \sum_{i=0}^{N-1} loss\left(y_i^{(s)}, p\left(l_k \middle| X_i^{(s)}, \theta^{(s)}\right)\right), \tag{5.5}$$

where

$$loss\left(y_i^{(s)}, p\left(l_k \middle| X_i^{(s)}, \theta^{(s)}\right)\right) = \sum_{k=0}^{N_c-1} -\log\left(p\left(l_k | X_i^{(s)}, \theta^{(s)}\right)\right) \cdot \delta(y_i = l_k) \tag{5.6}$$

is the negative log likelihood of the labels. The minimization of the negative log likelihood is equivalent to the minimization of the cross entropy between the empirical probability distribution defined by the training labels and the probability distribution defined by the model. The parameters were optimized via mini-batch stochastic gradient descent, using gradients computed via backpropagation. Adaptive moment estimation (Adam) [42], a commonly used adaptive learning rate optimization algorithm, was used as optimizer with a learning rate of 1e-3 and a mini-batch size of 64 trials.

The training phase was divided into two steps [43]. During the first training step (800 maximum number of epochs), the CNN was trained until the validation loss reached its minimum, performing early stopping. The training loss recorded at the first run minimum became the target threshold for the second run. During the second training step (800 maximum number of epochs), the validation set was included in the training set and the optimization continued until the validation loss reached the threshold loss.

This trialwise training strategy was applied to the basal Sinc-ShallowNet (Table 5.1), and all its variants (Table 5.2) on both ME- and MI-EEG dataset, to test the effect of different design choices on the decoding accuracy (see Section *Design choices*). Moreover, this strategy was applied to the three re-implemented SOA CNNs on both datasets, for a comparison with Sinc-ShallowNet performance (see Section 5.2.6), as well as to evaluate how the two training strategies affect different CNN architectures (see Sections *Cropped training strategy* and 5.2.6).

Cropped training strategy

Schirrmeister et al. [14] introduced a cropped training strategy for EEG decoding: they used crops of trials (i.e., sliding time windows within the trial) as input for the CNNs instead of the entire trial and set the target label of each crop equal to the label of the trial the crop belonged to. This leads to an augmented dataset that could increase the performance on the test set (i.e., additional regularizer effect). Actually, Schirrmeister et al. [14] reported a statistically significant improvement of cropped training only for deep architectures. Here, cropped training was applied to Sinc-ShallowNet (in its basal version), as well as to the re-implemented SOA CNNs, to compare trialwise training with cropped training for each network, in order to further study the effect of cropped training depending on the CNN architecture. To perform cropped training and allow a strict comparison with results of Schirrmeister et al. [14], the pre-processing of the MI dataset had to be modified by epoching signals between 0.5-4.0 s to keep the same epoching procedure as in [14] (i.e., an epoching procedure that allows the extraction of a few overlapped crops of 2 s), resulting in EEG patterns of shape $(1,22,875)$ as input. This is at variance with the 0.5-2.5 s epoching of the MI dataset adopted here for the other analyses (since such epoching was in agreement with other studies [11,20,36], see also Section *Motor imagery: BCI-IV2a dataset*). Therefore, for each CNN, the trialwise training on the MI dataset had to be performed also with the 0.5-4 s epoching to evaluate the effect of cropped training against trialwise training. Crops of 2 s (corresponding to 500 time samples) with a stride of 0.5 s (corresponding to 125 time samples) were extracted for each trial and these crops represented the CNN inputs. For each subject, this cropping procedure resulted in 6 crops $(1,44,500)$ per trial for the ME-EEG signals and 4 crops $(1,22,500)$ per trial for the MI-EEG signals, augmenting the available dataset. Adopting this training strategy, the CNNs output one prediction for each crop and thus several crop predictions belong to the same trial. To further regularize CNNs trained with cropped training, the same loss function designed by Schirrmeister et al. [14], named "tied sample loss function" (Equation 5.7) was employed. In particular, the cross-entropy of two neighbouring crop predictions is added to the usual negative log likelihood of the labels to drive the optimization towards more stable features across crops. Let us denote with t_c the start frame of the c-th crop, with T the crop size (i.e., number of crop temporal samples) and with $X_{i,c}^{(s)} = X_i^{(s)}[:,:,t_c:t_c+T]$ the c-th crop ($0 \leq c \leq 5$ and $0 \leq c \leq 3$ for the ME- and MI-EEG signals, respectively) belonging to the i-th trial of the s-th subject. Hence, the loss was modified to depend also on the prediction for the next crop $c+1$:

$$loss\left(y_i^{(s)}, p\left(l_k \middle| X_{i,c}^{(s)}, \theta^{(s)}\right)\right) = \sum_{k=0}^{N_c-1} -\log\left(p\left(l_k \middle| X_{i,c}^{(s)}, \theta^{(s)}\right)\right) \cdot \delta(y_i = l_k) + \sum_{k=0}^{N_c-1} -\log\left(p\left(l_k \middle| X_{i,c}^{(s)}, \theta^{(s)}\right)\right) \cdot p\left(l_k \middle| X_{i,c+1}^{(s)}, \theta^{(s)}\right). \quad (5.7)$$

Except for the loss function, cropped training follows the setting adopted for the trialwise training, sharing the same hyper-parameters (e.g., same optimizer, regularizers, learning rate, mini-batch size, etc.) and the same two-runs training procedure. Cropped training was applied to Sinc-ShallowNet (its basal version, see Table 5.1) and to the other three re-implemented CNNs.

5.2.5. Regularization

In addition to early stopping and cropped training which act as regularizers, other regularizing techniques were used and implicitly integrated in Sinc-ShallowNet, as specified in its description (see Sections *Block 1: Spectral and spatial feature extraction*, *Block 2: Aggregation*, *Block 3: Classification*). These are highlighted here:

i. Dropout [40]. This technique randomly sets the outputs of the previous layer to zero with a probability p, during each training update. This helps to prevent co-adaptation (i.e., that some neurons are highly dependent to others) which could lead to overfitting. In the proposed network, dropout with $p = 0.5$ was introduced in block 2 immediately after the average pooling layer.

ii. Batch normalization [38]. This technique mitigates a phenomenon named "internal covariate shift", i.e., the change in the distribution of the layers' activation due to the change of the trainable parameters during training [38]. This phenomenon hinders the learning since the layers continuously need to adapt to the changed distribution while training and is particularly severe in deep neural networks. Batch normalization reduces the internal covariate shift, and consequently accelerates the training, by normalizing the output feature maps of intermediate layers to zero mean and unit variance across each training mini-batch. This technique introduces two trainable parameters since the normalization is followed by a channelwise affine transformation (that serves to maintain the expressive power of the network), whose parameters of scaling and shift are learned during training. Batch normalization enables higher learning rates without the risk of divergence, reduces the influence of a specific initialization scheme on the training, and also regularizes the model [38]. While this technique is commonly used in deep neural networks, also shallow neural networks adopting batch normalization have been proposed in the literature. In particular, shallow CNNs including batch normalization have been recently applied to EEG signals for ME and MI decoding tasks [11,14], and for P300 detection [44]. Importantly, Schirrmeister et al. [14] reported an improved performance both in their shallow and deeper neural networks when using batch normalization compared to omitting it. Motivated by these previous results, we adopted this technique in our shallow CNN (blocks 1, 2) too, by applying it to the output of the convolutional layer immediately before the non-linearity, as recommended in the original paper [38], with a momentum term of $m = 0.99$ and with $\varepsilon = 1e - 3$ for numerical stability.

iii. Kernel max-norm regularization. This technique constraints the norm of the trainable parameters to be upper bounded by a fixed constant c. Typically, it improves the performance of mini-batch stochastic gradient descent training and it was found to be especially useful with dropout [40]. This technique was applied to the spatial depthwise convolutional (block 1) and to the fully-connected (block 3) layers similarly to [11], using $c = 1$ and $c = 0.5$, respectively.

These regularization techniques were also used in the other re-implemented CNNs, as proposed in their original formulation.

5.2.6. Classification performance and comparison with state-of-the-art approaches

The performance of Sinc-ShallowNet in its basal form (Table 5.1) was compared to the five variants (Table 5.2) and to the re-implemented SOA algorithms. The latter comprise three CNNs (EEGNet [11], DeepConvNet and ShallowNet [14]) and one traditional machine learning approach (FBCSP [45]+rLDA).

The three SOA CNNs (more details in Section 5.6.1 of Supplementary Materials) include different convolutional modules, while keeping a single fully-connected layer in the classification module. EEGNet consists of three convolutional layers (one of them depthwise and one separable), DeepConvNet of five convolutional layers, and ShallowConvNet of two convolutional layers. The first two CNNs are general-purpose architectures; the last CNN is designed specifically for oscillatory signal classification, learning features related to log band-power by the introduction of a squaring nonlinearity, an average pooling layer and a log nonlinearity after the convolutional module. As EEGNet was designed for 128 Hz EEG signals [11], we multiplied the lengths of its temporal kernels and pooling sizes by a scaling factor of 2 to learn features coherently with the sampling frequency used here (a similar procedure was adopted in [11] when previous CNNs were re-implemented for comparison with EEGNet). Then, as explained in Sections *Trialwise training strategy* and *Cropped training strategy*, these CNNs were trained as Sinc-ShallowNet, with trialwise and cropped training strategies. Compared to Sinc-ShallowNet (in its basal form having 13828 and 5508 trainable parameters in case of ME- and MI-EEG signals, respectively), the other three CNNs (EEGNet, ShallowConvNet and DeepConvNet) have a total number of trainable parameters of 2604, 82564, 298229 in case of ME-EEG signals, and of 1932, 40644, 278079 in case of MI-EEG signals, respectively. EEGNet and ShallowConvNet are both shallow architectures, the first one having an extremely low number of trainable parameters due to the low number of temporal kernels adopted in the first layer ($K_1 = 8$) and the use of depthwise and separable convolutions. These two architectures were chosen as reference shallow architectures (both general-purpose and specific for sensorimotor rhythm classification) to be compared with Sinc-ShallowNet. DeepConvNet was chosen as reference deep architecture (general-purpose) to be compared with Sinc-ShallowNet.

The traditional decoding pipeline adopted included FBCSP – a commonly used algorithm in EEG decoding and the winner of the BCI competition IV datasets 2a and 2b – coupled with rLDA. More details about the implementation of FBCSP+rLDA can be found in Section 5.6.2 of Supplementary Materials. This algorithm was used as the best-performing traditional approach in movement-related EEG decoding to be compared with Sinc-ShallowNet.

We adopted the decoding accuracy as performance metric of the classifiers; furthermore, for completeness, the confusion matrices of basal Sinc-ShallowNet and the benchmark traditional approach FBCSP+rLDA were computed. Wilcoxon signed-rank test was used to check for a statistically significant difference between the contrasted conditions. To correct for multiple tests, a false discovery rate correction at $\alpha = 0.05$ using the Benjamini-Hochberg procedure [46] was applied.

5.2.7. Interpretation

Post-hoc interpretation techniques were applied to Sinc-ShallowNet (in its basal version) at the end of the optimization. These include temporal and spatial kernel visualizations and an additional gradient-based technique, denoted as "temporal sensitivity analysis" (since it is applied to the features learned by the temporal sinc-convolutional layer).

Temporal and spatial kernels visualization

The visualization of the learned kernels of the first block allows the interpretation of the temporal and spatial convolutional layers. The temporal sinc-convolutional layer introduced in the Sinc-ShallowNet architecture allows a direct interpretation of the learned parameters, which are the lower and upper cutoff frequencies $f_{1,j}$ and $f_{2,j}$ of the K_1 band-pass filters. Hence, for each subject, the distribution of the learned temporal kernels can be visualized by displaying how their passbands are distributed within the frequency range of the input signals (i.e., (4,125] Hz for ME- and (4,38] Hz for MI-EEG signals), and the preferred EEG rhythm (e.g., α, β, etc.) can be immediately derived. In particular, the following EEG bands b were considered: θ = (4,8] Hz, α = (8,12] Hz, β = (12,30] Hz, low γ = (30,50] Hz, high γ = (50, 125] Hz. A temporal filter was considered belonging to a specific band b if its central frequency fell within that band (actually, in most cases the band-pass filters had narrow passbands totally falling within a specific band range, see also Section 5.3.3 in Results).

Moreover, since the spatial depthwise convolution applies separate spatial kernels to each temporally-filtered version of the input, the learned spatial kernels can be interpreted as the spatial features associated to a specific band-pass filter and can be visualized as scalp maps. Since we were interested in the evaluation of the discriminant power at the level of single electrode, here the absolute spatial kernel values were considered, as done by [9]. This visualization was limited to the spatial filters related to the more relevant and more class-specific band-pass filters (selected as described in Section *Temporal sensitivity analysis*).

Temporal sensitivity analysis

The visualization of the learned band-pass filters (see Section *Temporal and spatial kernels visualization*) provides information about their frequency-range preference but does not provide any information about their importance for the classification task. Hence, in order to quantify the relevance of the band-pass filters for the classification task, we designed the temporal sensitivity analysis inspired by the saliency maps [25]. This analysis allows the quantification of the importance of the different temporal kernels based on the gradient values, as described in the following (for simplicity, here we omit the superscript *s* referring to the specific subject).

i. Gradient computation. Given a class k of interest and the i-th test trial of the s-th subject $X_i \in \mathbb{R}^{C \times T}$ as input, let $Y_j \in \mathbb{R}^{C \times T_1}$ ($Y_{i,j}$when X_i is fed as input) be the output of the j-th temporal kernel (i.e., the j-th feature map) of the sinc-convolutional layer and $z_k = h_k(X;\theta) \in \mathbb{R}^{N_c}$ ($z_{i,k}$ when X_i is fed as input) be the class score (i.e., output of the block 3 fully-connected layer, immediately before the softmax activation function). The class score z_k is a highly non-linear function of Y_j; given the input test trial X_i, this function

can be approximated by a linear function in the neighbourhood of $Y_{i,j}$ by computing the first-order Taylor expansion [25] (Equation 5.8):

$$\begin{cases} z_k = z_k(Y_j) \approx G_{i,j,k}^{*T} \cdot Y_j^* + b_{i,j,k} \\ G_{i,j,k}^* = \left.\frac{\partial z_k}{\partial Y_j}\right|_{Y_{i,j}^*} \end{cases} . \quad (5.8)$$

In the Equation 5.8, the superscript $*$ denotes a vectorized form (column vector), superscript T represents the transposition of the vector, and $b_{i,j,k}$ a bias term. In this linearized expression, the magnitude of each element of $G_{i,j,k}^*$ quantifies how much the corresponding spatio-temporal sample within the j-th feature map (i.e., the j-th temporally filtered version of the input trial) affects the score for the k-th class z_k when presenting the input X_i. In other words, this quantifies how the value of an output category (e.g., output of the neuron related to class "Right Hand") changes with respect to a small change in the temporally filtered EEG signals.

ii. Gradient processing

a) For each $G_{i,j,k}$ (i.e., $\forall\, i,j,k$), the absolute value $|G_{i,j,k}|$ was computed and averaged across the spatial and temporal dimension to obtain a scalar value $\overline{|G_{i,j,k}|}$.

b) Quantities $\overline{|G_{i,j,k}|}$ related to trials belonging to each specific class were averaged together, resulting in the absolute gradient value $g_{j,k}$ (scalar value):

$$g_{j,k} = \frac{1}{M_k} \sum_i \overline{|G_{i,j,k}|}. \quad (5.9)$$

In Equation 5.9, the sum runs over the M_k trials belonging to the class k, i.e., $\{i : y_i = k\}$. Hence, $g_{j,k}$ quantifies how much, on average, the j-th temporal filter affects the score of the class k.

c) The gradients $g_{j,k}$ (Equation 5.9) were normalized dividing by the maximum across the classes and kernels (Equation 5.10):

$$\hat{g}_{j,k} = \frac{g_{j,k}}{\max\limits_{j,k} g_{j,k}} . \quad (5.10)$$

This was done in order to facilitate the comparison across kernels and classes.

Then, the normalized gradients $\hat{g}_{j,k}$ from Equation 5.10 were further processed in two ways for different purposes (d.1 and d.2).

d.1) Temporal sensitivity analysis at the level of EEG bands – For each considered EEG band b, $\hat{g}_{j,k}$ were averaged across the band-pass filters belonging to a specific EEG band b (see Section *Temporal and spatial kernels visualization*). The resulting score $\hat{g}_{b,k}$ (Equation 5.11) quantifies the overall importance of the specific band b for the classification of the specific class k:

$$\hat{g}_{b,k} = \frac{1}{K_{1,b}} \sum_j \hat{g}_{j,k}. \quad (5.11)$$

In Equation 5.11, the sum runs over the $K_{1,b}$ band-pass filters belonging to the b band, i.e., $\left\{j : f_{c,j} = \frac{f_{1,j}+f_{2,j}}{2} \in (f_{1,b}, f_{2,b}]\right\}$, where $(f_{1,b}, f_{2,b}]$ denotes the frequency range of the band.

d.2) Temporal sensitivity analysis at the level of single band-pass filter – This step was introduced to select the more relevant and more class-specific band-pass filters (i.e.,

the filters that are relatively more discriminative for a specific class than for the other classes) and to limit the visualizations of the learned spatial features to these selected temporal filters. Indeed, the normalized gradients $\hat{g}_{j,k}$ (Equation 5.10) corresponding to a specific temporal filter, can assume large values across all classes, indicating a large importance in the use of that temporal filter shared across the classes. To emphasize the specificity of each filter for a single class or a subset of classes, the gradient $\hat{g}_{j,k}$ was rescaled. The rescaling (Equation 5.12) was designed so that a gradient resulting higher (or lower) for a specific class than for the other classes on average, was scaled more (or less). This way, the differences of the filter relevance across the classes were emphasized:

$$\begin{cases} \hat{g}'_{j,k} = \gamma_{j,k} \cdot \hat{g}_{j,k}, \\ \gamma_{j,k} = \frac{3 \cdot \hat{g}_{j,k}}{\sum_{m=0, m \neq k}^{3} \hat{g}_{j,m}}. \end{cases} \tag{5.12}$$

Based on this scaling, the quantity $\hat{g}'_{j,k}$ assumes larger values ($\gamma_{j,k} > 1$) when the impact of j-th temporal filter on the score of the specific class k is higher than its average impact on the other three classes; vice versa it assumes lower values ($\gamma_{j,k} < 1$) when the j-th temporal filter impacts on average more on the other three classes than on the considered k class. Therefore, given a class k, filters having $\hat{g}'_{j,k} > \hat{g}_{j,k}$ (i.e., with $\gamma_{j,k} > 1$) represent the filters having a discriminative power relatively heavier for that class than for the other classes on average. Thus, considering a class k, the more relevant and more class-specific temporal band-pass filters can be identified as the filters with $\gamma_{j,k} > 1$ and that scored higher $\hat{g}'_{j,k}$ values. Lastly, the spatial kernels associated with the so selected band-pass filters can be visualized as described in Section *Temporal and spatial kernels visualization*.

5.3. RESULTS

5.3.1. Classification performance and comparison with state-of-the-art approaches

In this section, the performances of the basal Sinc-ShallowNet (trained via trialwise strategy) are compared with the traditional machine learning algorithm and with the three re-implemented CNNs (trained via trialwise strategy).

Figure 5.3 reports the confusion matrices obtained with the proposed architecture and with the machine learning algorithm FBCSP+rLDA, with ME- and MI-EEG signals. Each of these matrices represents the confusion matrix across the subject-specific classifiers. Denoting with i and j the i-th row and j-th column, the entry in the (i,j) location represents the total number of test trials across subjects predicted as class i when the true class is j (together with the % ratio between this number and the total number of trials for each class j). For each (i,j) location (16 in total), a Wilcoxon signed-rank test was performed between the entries of the subject-specific confusion matrices obtained with FBCSP+rLDA and with Sinc-ShallowNet, separately for the two datasets; that is, for each (i,j) location, we compared two samples of 14 values in case of the ME dataset and two samples of 9 values in case of the MI dataset. In order to correct for multiple comparisons (16 in total within each dataset), the Benjamini-Hochberg procedure was applied. The corrected p-value resulting from each comparison is displayed inside the corresponding cell of the matrices reporting Sinc-ShallowNet results (matrices on the right in Figure 5.3).

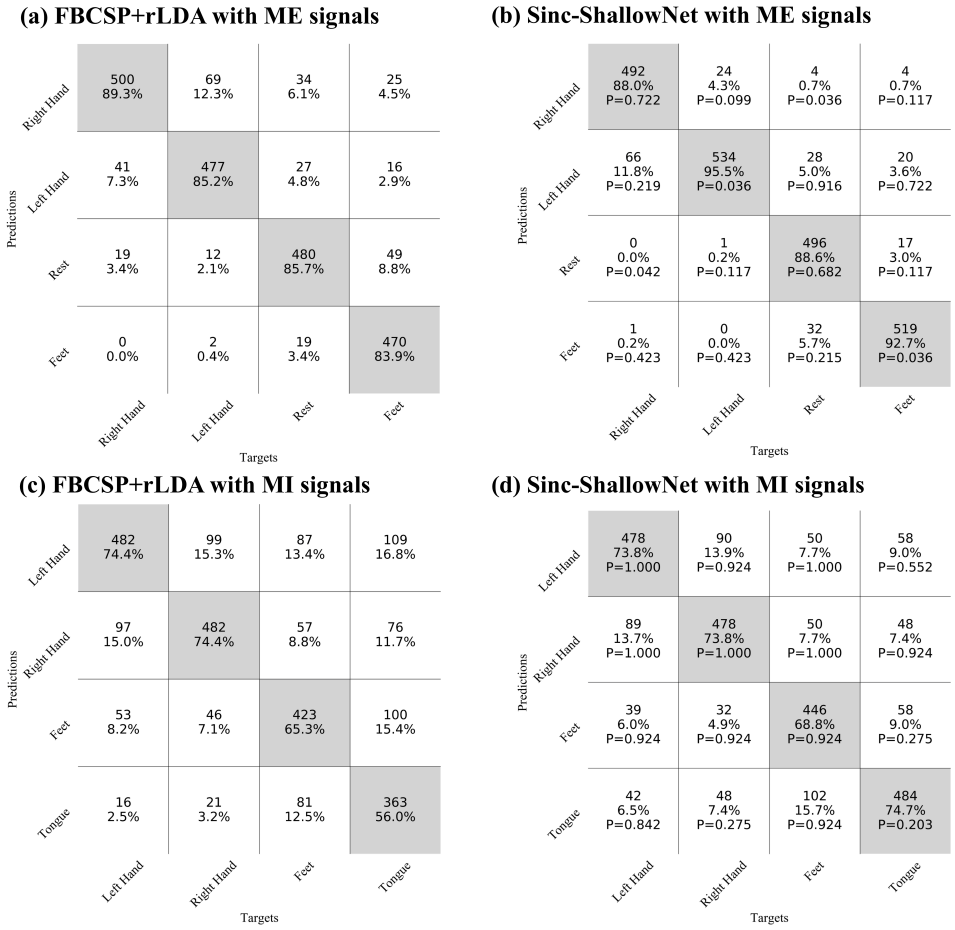


Figure 5.3 – Confusion matrices of FBCSP+rLDA ((a) and (c)) and of Sinc-ShallowNet ((b) and (d)). Sinc-ShallowNet was trained with trialwise strategy (see Section *Trialwise training strategy*). Matrices (a) and (b) were

computed across subject-specific classifiers on ME-EEG signals belonging to the test set, while (c) and (d) were computed on MI-EEG signals belonging to the test set. Each cell contains the total number of trials across subjects given a specific prediction and target label, and the ratio between this number and the total number of trials for each target label. For each (i,j) location (16 in total) of the confusion matrix (predicted class i, true class j), a Wilcoxon signed-rank test was performed between the entries of the subject-specific confusion matrices obtained with FBCSP+rLDA and with Sinc-ShallowNet, separately for the two datasets. Correction for multiple comparisons was obtained via the Benjamini-Hochberg procedure. The corrected p-value resulting from each comparison is displayed inside the corresponding cell of the matrices reporting Sinc-ShallowNet results.

The confusion matrices were similar between the approaches, with only 4 entries significantly different ($P < 0.05$) in case of ME-EEG signals. In particular, Sinc-ShallowNet classified significantly better "Left Hand" and "Feet" classes ($P = 0.036$) and produced a significantly lower number of misclassifications between "Right Hand" and "Rest" classes. In both algorithms, the majority of the misclassifications were associated with a wrong discrimination between "Right Hand"-"Left Hand" classes (110 misclassified trials for FBCSP+rLDA and 90 for Sinc-ShallowNet) in case of ME-EEG signals, and between "Right Hand"-"Left Hand" classes (196 misclassified trials for FBCSP+rLDA and 179 for Sinc-ShallowNet) and "Feet"-"Tongue" classes (181 misclassified trials for FBCSP+rLDA and 160 for Sinc-ShallowNet) in case of MI-EEG signals.

Tables 5.3 and 5.4 show the accuracies obtained with Sinc-ShallowNet, the three SOA CNNs, and the algorithm FBCSP+rLDA on ME- and MI-EEG signals, respectively. Results of the statistical analyses are reported too.

Table 5.3 – Accuracies (mean ± std across subjects) of the basal Sinc-ShallowNet and SOA algorithms, obtained with ME-EEG signals belonging to the test set. Here, the trialwise training was adopted. For each CNN, the total number of trainable parameters is reported in brackets. The corrected P values are reported (P_1 for each CNN vs. FBCSP+rLDA, P_2 for Sinc-ShallowNet vs. each SOA CNN).

Algorithm	Accuracy (%)	P_1	P_2
FBCSP+rLDA	86.0±9.0		
EEGNet (2604)	88.5±11.0	0.158	0.158
DeepConvNet (298229)	88.4±8.8	0.158	0.026
ShallowConvNet (82564)	93.9±9.3	0.024	0.040
Sinc-ShallowNet (13828)	91.2±9.1	0.024	

Table 5.4 – Accuracies (mean ± std across subjects) of the basal Sinc-ShallowNet and SOA algorithms, obtained with MI-EEG signals belonging to the test set. Here, the trialwise training was adopted (signal epoching 0.5-2.5 s). For each CNN, the total number of trainable parameters is reported in brackets. The corrected P values are reported (P_1 for each CNN vs. FBCSP+rLDA, P_2 for Sinc-ShallowNet vs. each SOA CNN).

Algorithm	Accuracy (%)	P_1	P_2
FBCSP+rLDA	67.5±13.9		
EEGNet (1932)	66.0±13.1	0.575	0.027
DeepConvNet (278079)	50.5±19.6	0.031	0.027
ShallowConvNet (40644)	71.6±14.2	0.046	0.302
Sinc-ShallowNet (5508)	72.8±12.9	0.031	

The proposed architecture scored an accuracy across subjects (mean ± std) of 91.2±9.1 % (inferior only to ShallowConvNet) and of 72.8±12.9 % (best overall) on ME- and MI-EEG

signals, respectively. Compared to the baseline FBCSP+rLDA algorithm, ShallowConvNet and Sinc-ShallowNet performed significantly better on both ME- ($P = 0.024$, $P = 0.024$, respectively) and MI-EEG signals ($P = 0.046$, $P = 0.031$, respectively). Sinc-ShallowNet significantly outperformed DeepConvNet ($P = 0.026$) on ME-EEG signals, and both EEGNet ($P = 0.027$) and DeepConvNet ($P = 0.027$) on MI-EEG signals. Lastly, ShallowConvNet significantly outperformed Sinc-ShallowNet ($P = 0.040$) on ME-EEG signals; however, regarding this point, further considerations can be drawn from the results of the post-hoc hyper-parameter evaluation (see Section 5.4.2 in the Discussion).

5.3.2. Post-hoc hyper-parameter evaluation and training strategy evaluation

The performance obtained with the basal Sinc-ShallowNet with ME- and MI-EEG signals was compared to the Sinc-ShallowNet variants, obtained by changing the hyper-parameters K_1, D_2, F_p, S_p and by introducing an additional pointwise convolutional layer as first layer in block 2 (see Section *Design choices*). Specifically, each variant was obtained by changing one hyper-parameter at a time while keeping the other hyper-parameters unchanged (see Table 5.2). In this comparison, both the basal Sinc-ShallowNet and each variant were trained adopting the trialwise training strategy (see Section *Trialwise training strategy*). The overall effect of each hyper-parameter change was quantified jointly on ME- and MI-EEG signals by computing the difference in accuracy between the tested (variant) and basal configurations $\Delta_{acc} = acc_{tested} - acc_{ref}$ (e.g., $\Delta_{acc} = acc_{K_1=8} - acc_{K_1=32}$ for the comparison "$K_1 = 8$–$K_1 = 32$", contrasting the configuration with $K_1 = 8$ temporal filters and the basal configuration having $K_1 = 32$ filters). The results are shown in Figure 5.4a: a significant worsening of the performance occurred when K_1 decreased ($P = 0.005$ and $P = 0.010$ when comparing $K_1 = 8$ vs $K_1 = 32$ and $K_1 = 8$ vs $K_1 = 16$, respectively), while no significant effect was induced by the other hyper-parameter changes.

We evaluated the impact of cropped training compared to trialwise training on Sinc-ShallowNet (in its basal configuration) and on each re-implemented SOA CNNs. As detailed in Section *Cropped training strategy*, the trialwise training strategy adopted for this analysis was designed with a different epoching of the MI-EEG signals (0.5-4 s rather than 0.5-2.5 s as adopted in the rest of the presented results) in order to follow the procedure used in [14]. Nevertheless, we verified that no statistically significant difference in performance emerged between the trialwise training implemented with the different epoching of MI-EEG signals ($P = 0.441, P = 0.345, P = 0.347, P = 0.346$, respectively for DeepConvNet, ShallowConvNet, Sinc-ShallowNet and EEGNet.). The overall effect of cropped training on each CNN was quantified jointly on ME- and MI-EEG signals by computing the difference in accuracy between the cropped and the trialwise training strategies $\Delta_{acc} = acc_{cropped} - acc_{trialwise}$. The corresponding results are shown in Figure 5.4b. Only the deep architecture DeepConvNet significantly benefited from the cropped training strategy ($P = 0.002$), while shallower architectures such as Sinc-ShallowNet and EEGNet performed significantly worse when trained with the cropped strategy ($P = 0.008$ and $P = 0.009$).

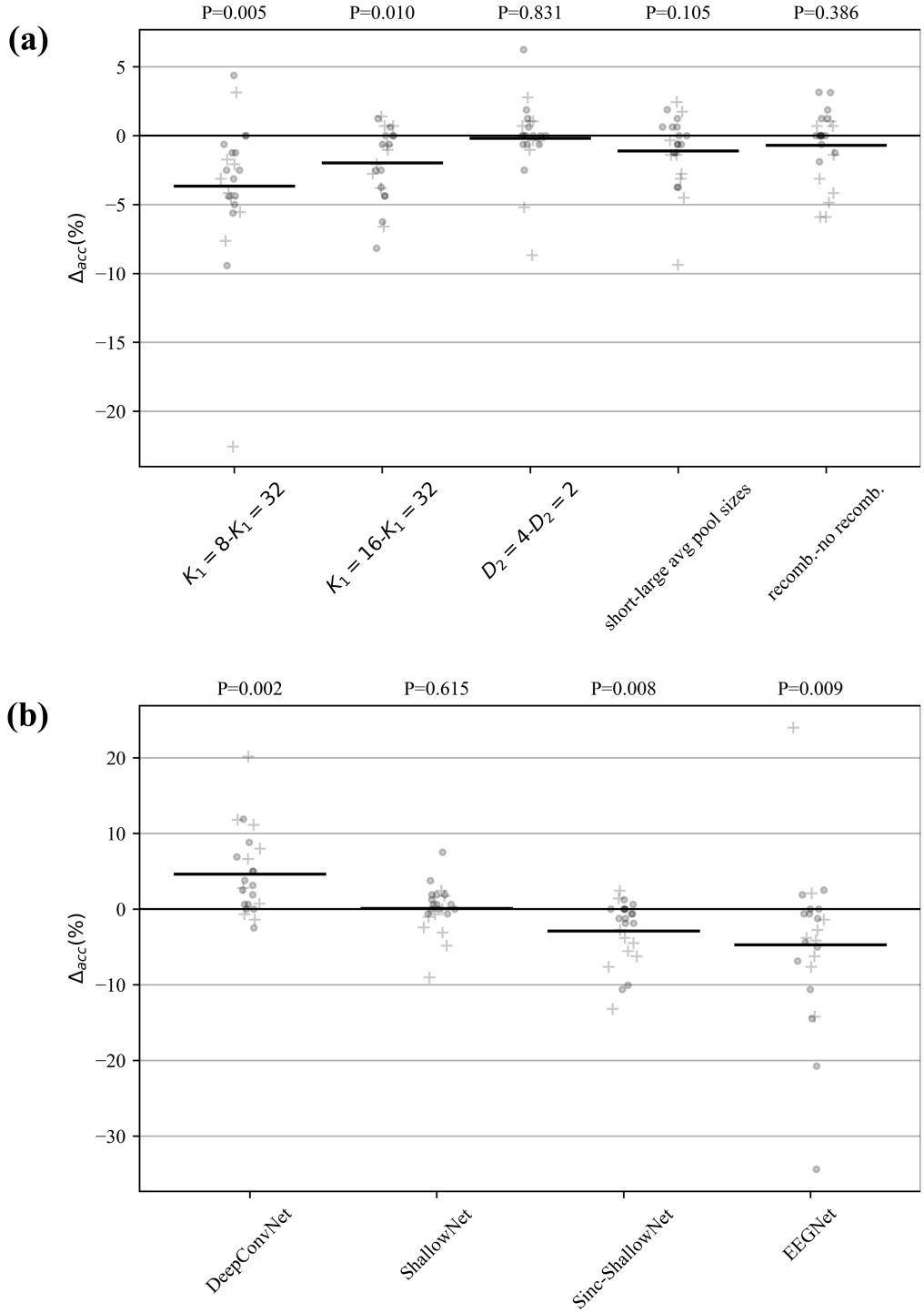


Figure 5.4 – Results of the analyses on Sinc-ShallowNet design choices and on training strategies. (a) Effect of the changes in the hyper-parameters of Sinc-ShallowNet (see Table 5.2) on the performance metric. The changes in accuracy (Δ_{acc}) were computed as the difference between the tested and the reference (i.e., basal) configuration ($\Delta_{acc} = acc_{tested} - acc_{ref}$, e.g., $acc_{K_1=8} - acc_{K_1=32}$). (b) Effect of the two different training strategies applied to each SOA CNN and to Sinc-ShallowNet on the performance metric. The changes in accuracy (Δ_{acc}) were computed as the difference between the cropped and trialwise training strategies ($\Delta_{acc} = acc_{cropped} - acc_{trialwise}$). For this comparison, MI-EEG signals were epoched between 0.5 and 4 s (see Section *Cropped training strategy*). In both panels, Δ_{acc} obtained with ME-EEG signals (◦) and with MI-EEG signals (+) were grouped together. The corrected P values are reported (Sinc-ShallowNet vs. each variant, trialwise vs. cropped training).

5.3.3. Interpretation

In order to illustrate feature interpretability and feature relevance evaluation enabled by the proposed approach, we provide the results of the interpretation techniques for one representative subject for each dataset (ME- and MI-EEG signals). These results refer to the basal Sinc-ShallowNet trained with the trialwise training strategy.

Figures 5.5a and 5.6a display the distribution of the temporal filters learned by the network for a specific subject in case of the ME- and MI-EEG signals, respectively. Most of the temporal band-pass filters belonged to specific EEG bands (a filter is considered belonging to an EEG band based on its central frequency, see Section *Temporal and spatial kernels visualization*). The learned band-pass filters mainly belonged to the β, low γ and high γ bands in case of ME-EEG signals (Figure 5.5a) and to the α, β and low γ bands in case of the MI-EEG signals (Figure 5.6a). The corresponding gradients $\hat{g}_{b,k}$ (see Equation 5.11 in Section *Temporal sensitivity analysis*) obtained from the temporal sensitivity analysis at the level of EEG bands are displayed in Figures 5.5b and 5.6b. These visualizations suggest that the classification tasks rely differently on the EEG bands depending on the class. The high γ band resulted the most important EEG band for each class of ME-EEG signals (Figure 5.5b) in addition to the β band – for the "Right Hand" and "Left Hand" classes – and low γ band for the "Rest" and "Feet" classes. The β band resulted relevant for each class of MI-EEG signals (Figure 5.6b) in addition to the α band – in particular for the "Left Hand" but also for the "Right Hand" classes – and low γ in particular for "Tongue" and also for "Feet" classes.

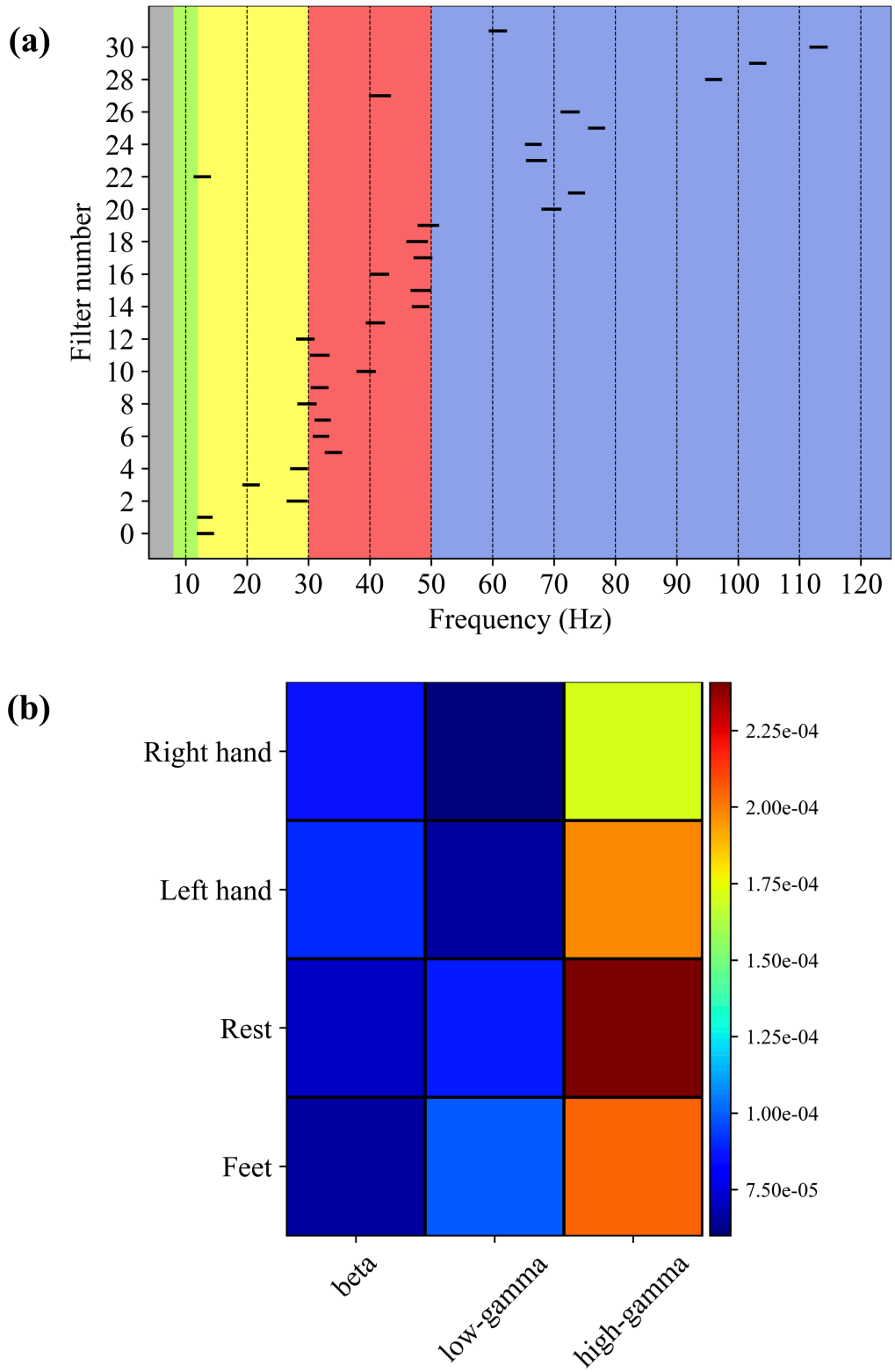


Figure 5.5 – Visualization and interpretation of the features learned by the temporal sinc-convolutional layer of Sinc-ShallowNet in case of ME-EEG signals of subject 12 (decoding accuracy 95.6%). (a) Visualization of the passband learned by each of the 32 filters. Each passband is displayed as a black line, with the end points representing $f_{1,j}$ and $f_{2,j}$ of the j-th learned filter. The colour-code used is: gray-θ, green-α, yellow-β, red-low γ, blue-high γ. (b) Results of the temporal sensitivity analysis at the level of EEG bands: the normalized gradient averaged across the band-pass filters belonging to a specific EEG band ($\hat{g}_{b,k}$) is displayed (colour-coded) for each class and each EEG band.

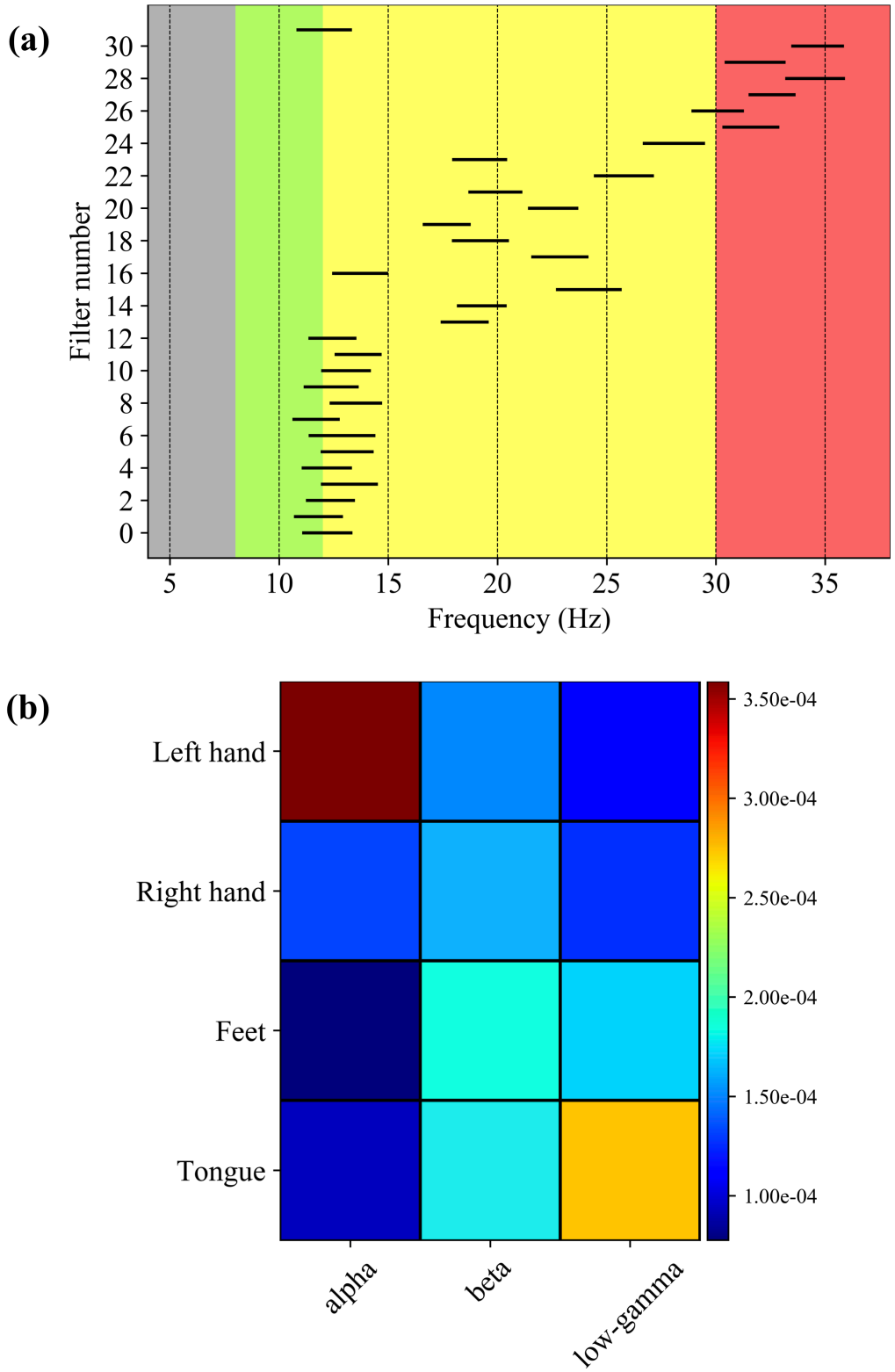


Figure 5.6 – Visualization and interpretation of the features learned by the temporal sinc-convolutional layer of Sinc-ShallowNet in case of MI-EEG signals of subject 3 (decoding accuracy 86.1%). (a) Visualization of the passband learned by each of the 32 filters. Each passband is displayed as a black line, with the end points representing $f_{1,j}$ and $f_{2,j}$ of the j-th learned filter. The colour-code used is: gray-θ, green-α, yellow-β, red-low γ. (b) Results of the temporal sensitivity analysis at the level of EEG bands: the normalized gradient averaged across the band-pass filters belonging to a specific EEG band ($\hat{g}_{b,k}$) is displayed (colour-coded) for each class and each EEG band.

Figures 5.7 and 5.8 report the results of the temporal sensitivity analysis performed at the level of the single band-pass filter for each decoded class, as to the same exemplary cases of Figures 5.5 and 5.6 (ME- and MI-EEG signals, respectively). In each panel (bar plot), both the normalized gradient $\hat{g}_{j,k}$ (Equation 5.10, length of the black line) and the rescaled gradient $\hat{g}'_{j,k}$ (Equation 5.12, length of the coloured bar), are displayed for each learned filter, together with the indication (colour-coded) of the band the filter belong to. By looking at $\hat{g}_{j,k}$, the filters belonging to each band assumed different importance depending on the class, in agreement with Figures 5.5b and 5.6b. For example, as to Figure 5.7, filters in the low γ band had on average larger values of $\hat{g}_{j,k}$ for the "Rest" and "Feet" classes than for the "Hand" classes. Moreover, within each class, filters in the high γ band had on average larger values of $\hat{g}_{j,k}$ compared to filters in the other bands, especially for the "Rest" and "Feet". However, by looking at the single filters, some of them had very similar gradient values $\hat{g}_{j,k}$ across all classes (for example filters #26, #28, #30 in Figure 5.7a, and filters #1, #7 in Figure 5.8b). The rescaled gradient $\hat{g}'_{j,k}$ allows the identification of the more relevant and more class-specific band-pass filters, as described in Section *Temporal sensitivity analysis*. Specifically, for each of the two more discriminative EEG bands (as obtained via the temporal sensitivity analysis at the level of EEG bands, Figures 5.5b and 5.6b), the two more relevant band-pass filters were selected as the two filters (belonging to that band) that scored the two highest values of $\hat{g}'_{j,k}$ with $\hat{g}'_{j,k} > \hat{g}_{j,k}$. For the so-selected temporal filters, the $D_2 = 2$ learned spatial filters were displayed as to their absolute values (insets within each panel of Figures 5.7 and 5.8). The blue regions correspond to weights that are around 0 indicating electrode locations with a low discriminant power, and vice versa for the red regions. Thus, spatial filters extremely focalized to specific subsets of electrodes were learned for both the decoding tasks. In particular, a clear contra-laterality in the scalp weight distributions can be observed in case of the hand movements (both executed and imagined) compared to the other classes.

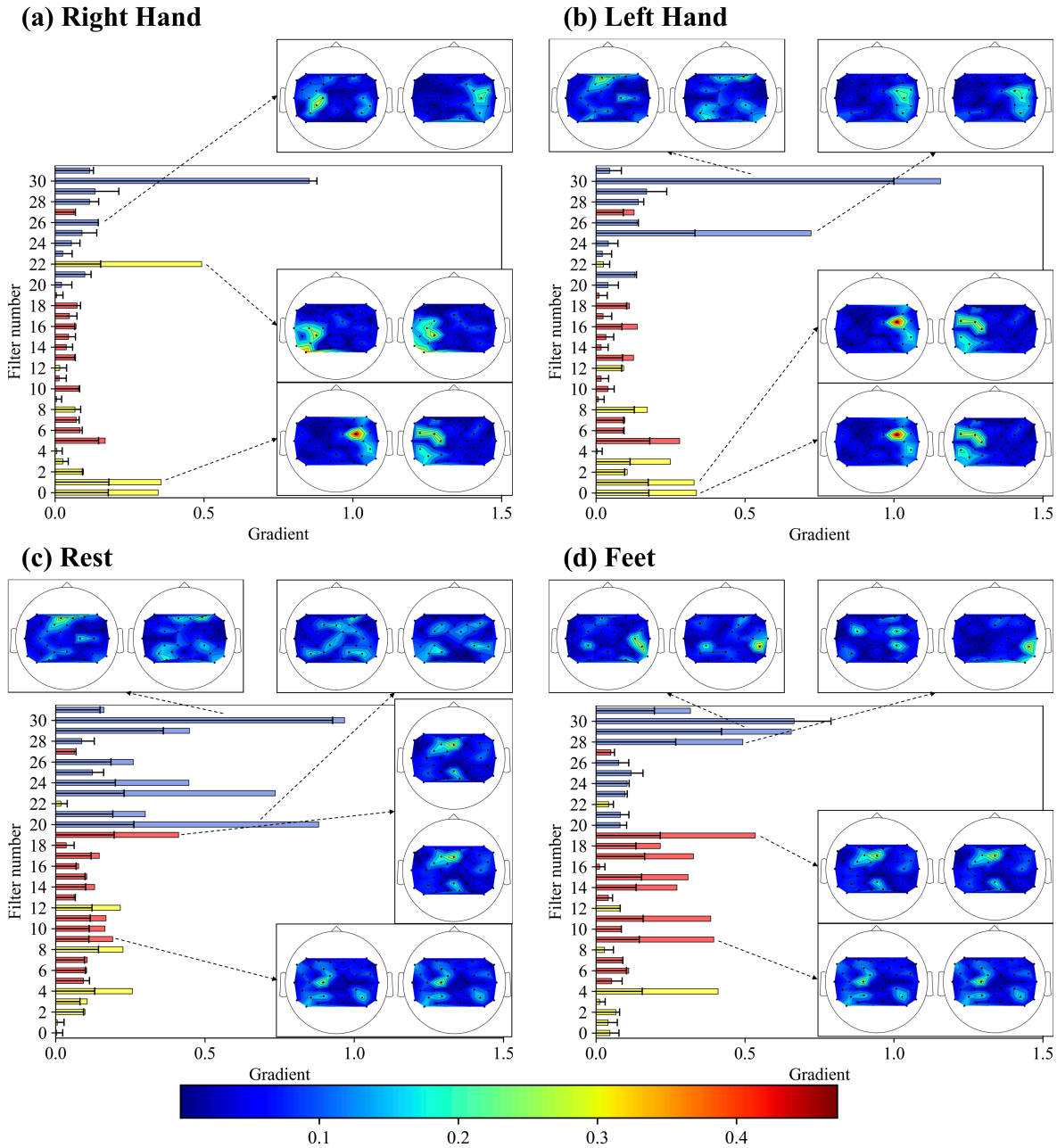


Figure 5.7 – Spatial distribution of the more relevant and more class-specific band-pass filters learned by Sinc-ShallowNet in case of ME-EEG signals of subject 12 (the same as in Figure 5.5). Each panel refers to a specific class (a-d for "Right Hand", "Left Hand", "Rest", and "Feet", respectively) and shows the results of the temporal sensitivity analysis at the level of each single band-pass filter by displaying both the normalized gradient ($\hat{g}_{j,k}$) and rescaled ($\hat{g}'_{j,k}$) gradient of the single filters for that specific class. The coloured bars denote the rescaled gradients (the colour indicates the EEG band the filter belongs to, i.e., gray-θ, green-α, yellow-β, red-low γ, blue-high γ), while the black lines denote the normalized gradients. The latter are reported in order to identify an increase in the rescaled gradients. For each class, the two more important band-pass filters within each of the two more important EEG bands (according to Figure 5.5b) are selected depending on the value of the increased rescaled gradients. For the so-selected band-pass filters, the spatial distribution is displayed by drawing the absolute values of the corresponding two spatial filters. In case of the "Right Hand" class, only one band-pass filter (#26) within the high γ band was selected for this visualization since it was the only one having $\hat{g}'_{j,k} > \hat{g}_{j,k}$.

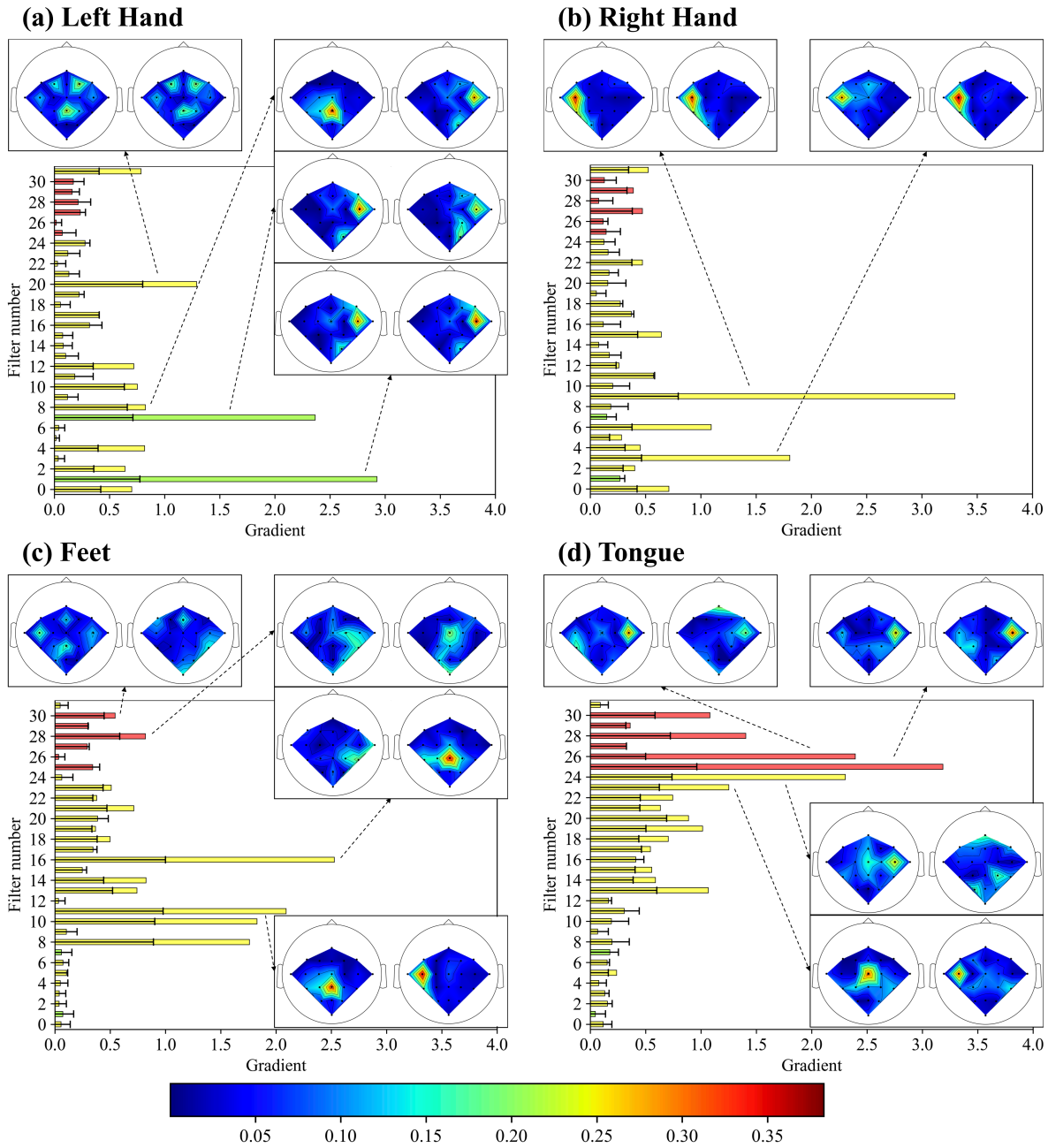


Figure 5.8 – Spatial distribution of the more relevant and more class-specific band-pass filters learned by Sinc-ShallowNet in case of MI-EEG signals of subject 3 (the same as in Figure 5.6). Each panel refers to a specific class (a-d for "Left Hand", "Right Hand", "Feet", and "Tongue", respectively) and shows the results of the temporal sensitivity analysis at the level of each single band-pass filter by displaying both the normalized gradient ($\hat{g}_{j,k}$) and rescaled gradient ($\hat{g}'_{j,k}$) of the single filters for that specific class. The coloured bars denote the rescaled gradients (the colour indicates the EEG band the filter belongs to, i.e., gray-θ, green-α, yellow-β, red-low γ), while the black lines denote the normalized gradients. The latter are reported in order to identify an increase in the rescaled gradients. For each class, the two more important band-pass filters within each of the two more important EEG bands (according to Figure 5.6b) are selected depending on the value of the increased rescaled gradients. For the so-selected band-pass filters, the spatial distribution is displayed by drawing the absolute values of the corresponding two spatial filters. In case of the "Right Hand" class, the band-pass filters within the α band (#1 and #7) were not selected for the visualization since $\hat{g}'_{j,k} < \hat{g}_{j,k}$ for these filters.

5.4. DISCUSSION

In this study Sinc-ShallowNet, a novel lightweight and interpretable CNN for EEG decoding, was designed and applied to motor execution and imagery tasks. The use of a band-pass filtering specialized convolutional layer (sinc-convolutional layer) and a spatial filtering with a reduced CNN channel connectivity (depthwise convolutional layer) enables the learning of band-pass filters and directly associated spatial filters. Thus, the proposed CNN is fully-interpretable and optimized in its convolutional module (i.e., feature extractor). In particular, the following points of strength can be emphasized:

i. Easy interpretation of both spectral and spatial features. The trainable parameters of the sinc-convolutional layer are directly interpretable (cutoff frequencies instead of mere kernel values as in a traditional convolutional layer) and the spatial filters are directly tied to specific band-pass filters.
ii. High optimization in terms of number of trainable parameters. The adopted sinc-convolution trains only 2 cutoff frequencies for each temporal filter and the depthwise convolution reduces the connections across the CNN channels.
iii. Computational efficiency. Due to the symmetry of the parametrized function adopted in the sinc-convolution, only half of the kernel values need to be computed.

In addition, the interpretation of the learned spectral and spatial features was further enriched thanks to the temporal sensitivity analysis; this analysis allows the identification of the more discriminative EEG bands (temporal sensitivity analysis at the level of EEG bands), and the more relevant and more class-specific band-pass filters (temporal sensitivity analysis at the level of single band-pass filter) together with their spatial distribution.

5.4.1. Classification performance and comparison with state-of-the-art approaches

The results on the ME and MI decoding tasks suggest that Sinc-ShallowNet significantly outperformed the traditional FBCSP+rLDA decoding pipeline. Among the re-implemented SOA CNNs, only ShallowConvNet (but not DeepConvNet and EEGNet) performed significantly better than the traditional machine learning approach, in agreement with results by Schirrmeister et al. [14].

By comparing Sinc-ShallowNet with the re-implemented CNNs, the following considerations can be drawn. First, ShallowConvNet significantly outperformed Sinc-ShallowNet on ME- but not on MI- EEG signals (see Table 5.3). This is the only case in which Sinc-ShallowNet performed worse compared to the other considered CNNs. Nevertheless, it is worth noticing that Sinc-ShallowNet introduces 13828 and 5508 trainable parameters, that corresponds only to the 16.7% and 13.6% of those introduced by ShallowConvNet in case of ME- and MI-EEG signals (82564 and 40644), respectively. Therefore, the proposed architecture finalized the classification tasks in a more computationally efficient way, by introducing a lower number of trainable parameters. Furthermore, ShallowConvNet architecture was developed specifically for sensorimotor rhythm classification forcing the extraction of log band-power features (task-specific CNN), while Sinc-ShallowNet was not restricted to specific feature learning. Second, in the comparison with a general-purpose shallow architecture (EEGNet), Sinc-ShallowNet performed significantly better on MI-EEG signals, while performed comparably on ME-EEG signals. The lower performance of EEGNet

may derive from the extremely lightweight architecture that used only $K_1 = 8$ temporal filters. Accordingly, the decoding of MI-EEG signals may benefit from a higher number of temporal filters (e.g., 32 as in the architecture proposed here). The introduction of the temporal sinc-convolutional layer that reduces the number of trainable parameters (i.e., only the two cutoff frequencies for each temporal filter) may be particularly beneficial for the decoding of MI-EEG dataset. Indeed, this dataset is characterized by a low number of training examples that requires the number of trainable parameters to be carefully maintained limited in order to avoid overfitting and achieve a good fit. Furthermore, when comparing Sinc-ShallowNet with DeepConvNet, the first provided significantly higher decoding accuracy on both ME- and MI-EEG signals. This may be attributable to the higher number of trainable parameters introduced by DeepConvNet (298229 and 278079 in case of ME- and MI-EEG signals, respectively), leading to an architecture more prone to overfitting especially in case of small datasets as for the adopted MI dataset.

5.4.2. Design choices of Sinc-ShallowNet

The post-hoc hyper-parameter evaluation (Figure 5.4a), revealed a significant negative effect of lowering K_1 on Sinc-ShallowNet performance, with an average $\Delta_{acc} = -4\%$ and $\Delta_{acc} = -2\%$, when using 8 and 16 band-pass filters compared to 32 filters, respectively. Thus, Sinc-ShallowNet benefits from an increased set of band-pass filters that enrich the temporally filtered representation of the input. Furthermore, Sinc-ShallowNet performance on both datasets when using $K_1 = 8$ was not different from EEGNet that uses this number of temporal filters.

The analysis on D_2 and on the optional recombination deserves some comments. Increasing D_2 did not lead to significant increase in the performance. However, it is interesting to note that when the effect of D_2 was disaggregated between the two datasets (ME-EEG and MI-EEG dataset), an opposite behavior tends to appear, with an average $\Delta_{acc} = +0.4\%$ and $\Delta_{acc} = -0.2\%$ on ME- and MI-EEG signals respectively (although not statistical significance was reached in either dataset). This different behavior might be explained considering that when a CNN is trained with EEG signals containing a lower number of frequency components (such as MI-EEG signals), the band-pass temporal filters lie into a narrower frequency range and thus the probability that two different temporal filters have similar cutoff frequencies is higher. In this scenario, a lower number of spatial filters (D_2) for each temporal filter could be sufficient to retain enough capacity of the CNN, because close temporal filters could compensate for the lower D_2. Indeed, different spatial filters could be learned for similar temporal filters obtaining a cumulative set (across similar temporal filters) of band-specific spatial filters. Conversely, ME-EEG signals having wider frequency content can benefit from a larger number D_2 of spatial filters. Recombining the spatial activations via an additional pointwise convolutional layer did not improve accuracy. However, in this case too, by disaggregating the effect on the two datasets, an opposite behavior tends to appear with an average $\Delta_{acc} = +0.5\%$ and $\Delta_{acc} = -2.6\%$ in case of ME- and MI-EEG signals respectively (although not statistical significance was reached in either dataset). This may be due to the learning of a useful recombination of frequency-specific spatial features learned across a wide frequency range, in case of signals with broad frequency content as ME-EEG signals. Finally, it is worth noticing that both

increasing D_2 and including a pointwise convolutional layer lead to an increase in the number of trainable parameters that might be critical in applications involving small datasets (e.g., the adopted MI dataset). Overall, these considerations remain quite speculative and further experiments are required, for example testing Sinc-ShallowNet and its different design choices on other datasets having larger and smaller size than those used here and having various frequency contents. However, it is interesting to note that the small accuracy increase ($\Delta_{acc}= +0.5\%$) in case of ME-EEG signals obtained introducing the pointwise convolutional layer led to a significant better performance of Sinc-ShallowNet compared to EEGNet ($P = 0.046$) and to comparable performance with ShallowConvNet ($P = 0.090$); at the same time, the accuracy decrease in case of MI-EEG signals ($\Delta_{acc}= -2.6\%$) did not change the statistical significance ($P = 0.049$ vs. EEGNet and DeepConvnet, $P = 0.340$ vs. ShallowConvNet). Thus, the proposed Sinc-ShallowNet architecture integrated with the recombination of the spatial activations led to a CNN that performs better than or at least as well as the SOA CNNs on both datasets, at the expense of the number of trainable parameters (17924 and 9604 in case of ME and MI datasets respectively).

Lastly, changing the average pooling strategy by using larger pool and stride sizes did not affect the performance.

In conclusion, this analysis suggests that the proposed Sinc-ShallowNet in its basal version (see Table 5.1) resulted in a good compromise between performance and parsimony with enough capacity to solve both the decoding tasks.

5.4.3. Training strategies

The overall effect of the training strategy on the performance metric (Figure 5.4b) resulted in a significantly increase of the decoding accuracy for a deeper architecture as DeepConvNet (on average $\Delta_{acc}= +4.6\%$), while a significant worsening of the performance was observed as the CNN architecture becomes shallower and more lightweight (no significant effect on ShallowConvNet, $\Delta_{acc}= -2.9\%$ for Sinc-ShallowNet and $\Delta_{acc}= -4.7\%$ for EEGNet on average). This different behavior of cropped training on shallow and deep architectures is in line with the results reported by Schirrmeister et al. [14] when examining ShallowConvNet and DeepConvNet, i.e., no improvements for ShallowConvNet and significant improvement for DeepConvNet. The present study further confirmed those previous results and extended them to other shallow architectures (i.e., EEGNet and Sinc-ShallowNet). Thus, a data-intensive CNN (e.g., DeepConvNet) improved its performance with cropped training – which acts as a data augmentation procedure – while lightweight CNNs did not. In contrast to deeper network, shallow CNNs like EEGNet and Sinc-ShallowNet performed well in both the decoding tasks without the need of any data augmentation procedure that, conversely, worsened their performance.

5.4.4. Interpretation

The band-pass filters mainly belonged to the β, low γ and high γ EEG bands when the network was trained with ME-EEG signals (Figure 5.5a), and to the α, β and low γ EEG bands when the network was trained with MI-EEG signals (Figure 5.6a). The latter result agreed with that obtained by Lawhern et al. [11] using EEGNet on the same decoded subject. In particular,

Lawhern et al. [11] estimated each band-pass filter learned by the temporal convolutional layer simply by counting the number of cycles of the specific temporal kernel in the corresponding temporal window. In Sinc-ShallowNet, each band-pass filter is implicitly defined by the temporal sinc-convolutional layer that directly provides the two cutoff frequencies.

When the CNN was trained on ME-EEG signals, the temporal sensitivity analysis at the level of EEG bands (Figure 5.5b) indicates that the most relevant bands were β, high γ for the "Right Hand" and "Left Hand" classes, and low γ, high γ for the "Feet" and "Rest" classes. In addition, the high γ band emerged as more important than the β and low γ bands for each decoded class, confirming the relevance not only of the β but also of the high γ band in the decoding task as previously evidenced by Schirrmeister et al. [14]. Ball et al. [27] found an increase in the high γ activity within the 60-90 Hz range, in addition to lower frequencies activity (α, β), in human sensorimotor cortex during ME. Interestingly, in the exemplary case shown in Figure 5.5, most of the band-pass kernels belonging to the high γ band fell within this range (7 out of 10).

When the CNN was trained on MI-EEG signals, the temporal sensitivity analysis at the level of EEG bands (Figure 5.6b) indicates that the most relevant bands were α, β for the "Left Hand" and "Right Hand" classes, and β, low γ for the "Feet" and "Tongue" classes. These results are in line with previous studies showing that also the low γ band, together with the α and β bands, provides information on MI [28]. This was further confirmed by [47], where adding low γ features to α and β features led to better performance using the same MI dataset.

Thanks to the use of spatial depthwise convolution, the proposed architecture ties spatial kernels to each band-pass filter and thus, the relevance, as quantified by the temporal sensitivity analysis, can be propagated from each band-pass filter to the associated spatial filters. In particular, the more relevant and more class-specific spatial filters can be identified – as those associated to the band-pass filters scored by the highest rescaled gradients $\hat{g}'_{j,k}$, (i.e., temporal sensitivity analysis at the level of single band-pass filter) – and visualized. These spatial filters show a highly localized distributions in the scalp maps (Figures 5.7a-5.7d and 5.8a-5.8d, respectively for ME- and MI-EEG signals). Among the spatial filters specific for the hand movements, some filters have the most discriminative electrodes located in the contralateral hemisphere to the executed and imagined hand movement, approximately above the primary sensorimotor hand representation areas (i.e., around C3 and C4). Regarding the executed and imagined feet movements, some filters have the most discriminative electrodes located more centrally, approximately above the primary motor foot area (i.e., around CPz, Cz and FCz). Finally, regarding the imagined tongue movement, the most discriminative electrodes are placed not only around C3 and C4, but also approximately above the somatosensory cortex (i.e., area below Cz), representing the brain region triggered by the imagination of tongue movements [18].

Therefore, by interpreting the features exploited by the network for the classification task, it turns out that Sinc-ShallowNet was capable of learning features related to known neurophysiological phenomena without relying on artefact or noise sources in the EEG signals.

As underlined previously, the interpretation capabilities of the network are provided by coupling an interpretable layer (sinc-convolutional layer) with an optimized layer (depthwise convolutional layer), and by using a post-hoc gradient-based technique alongside with spatial

and temporal filter visualizations. Therefore, interpretation capabilities of Sinc-ShallowNet are intrinsically linked to some specific design choices and specifically implemented post-hoc analyses. However, other more general-purpose techniques adopted in our network (e.g., batch normalization or dropout), that introduce a regularization effect, contribute to increase the neurophysiological reliability of feature interpretation by improving the performance on unseen examples. For example, we verified that when training Sinc-ShallowNet by removing the batch normalization layers in the blocks 1, 2 (and leaving all the other hyper-parameters unchanged), a significant decrease of the decoding accuracies occurred: $\Delta_{acc} = -4.8\%$ ($P = 0.002$, Wilcoxon signed-rank test), $\Delta_{acc} = -14.8\%$ ($P = 0.008$, Wilcoxon signed-rank test) respectively for ME- and MI-EEG signals, where $\Delta_{acc} = acc_{w/o\ BN} - acc_{w/\ BN}$. These simulations confirmed the important regularization introduced by batch normalization that significantly increased network accuracy on unseen examples. Accordingly, although batch normalization does not contribute directly to the interpretation capabilities of the network (omitting it the inner interpretation capabilities of the network are not altered), its inclusion increases the neurophysiological significance of the interpreted features via accuracy improvement. Indeed, the band-pass filters and spatial filters learned by the batch-normalized Sinc-ShallowNet turn out to be more class-discriminative (as they provide higher accuracies). Therefore, the learned spectral and spatial features are more likely to reflect neurophysiological aspects (in terms of more relevant EEG bands and electrodes) linked to the investigated tasks (i.e., motor execution and motor imagery decoding).

Finally, we would like to provide some comments on other CNNs in the literature that adopt a non-traditional convolutional layer designed to perform a specific input transformation (here the sinc-convolutional layer forcing band-pass filtering). First, it is worth noticing that, at best of our knowledge, only two previous (and very recent) studies [17,18] include a similar layer within a CNN architecture, indicating that this represents an innovative and emerging approach in the field of EEG decoding. Zhao et al. [18] proposed a CNN for MI classification including a time-frequency convolutional layer based on wavelets and interpreted the learned features. Differently from the architecture proposed here, they adopted a traditional spatial convolutional layer and tested the network only on MI decoding tasks. Comparing the decoding accuracy reported in the original paper [18] with Sinc-ShallowNet accuracy on the same MI-EEG signals, Sinc-ShallowNet scored an average accuracy $+5.8\%$ with respect to the architecture proposed by Zhao et al. [18], although without reaching statistical significance ($P = 0.086$, Wilcoxon signed ranked test). However, the network by Zhao et al. [18], due to the adoption of a standard spatial convolutional layer (that by itself involves 13775 trainable parameters, including bias), has a larger number of trainable parameters compared to Sinc-ShallowNet (1408 for MI-EEG signals). In an even more recent paper, Zeng et al. [17] included a sinc-convolutional layer into a deep 1D CNN (3 convolutional layers and 4 fully-connected layers) for EEG emotion classification. The proposed solution appears more robust and more performing than other classifiers (and thus possibly confirming the potentiality of this kind of layer). However, the network by Zeng et al. [17] introduced a large number of trainable parameters, especially due to the use of 3 hidden fully-connected layers having thousands of neurons. Moreover, the authors did not face the interpretation of the learned features; in particular, the adoption of a reshaped input representation (2D-to-1D reshaping) and of

traditional convolutions hinder the interpretability of the CNN. In future, it will be interesting to test Sinc-ShallowNet on the same decoding task tackled by Zeng et al. [17].

5.5. CONCLUSIONS

In conclusion, we proposed a novel CNN named Sinc-ShallowNet, characterized by an interpretable and efficient (in terms of number of trainable parameters) convolutional module. This module includes a temporal sinc-convolutional layer, forcing the learning of band-pass filters with only two trainable parameters per filter, and a spatial depthwise convolution that learns spatial features tied to each band-pass filter. The proposed design provides direct interpretability of the learned spectral-spatial features, at the same time limiting the number of trainable parameters. Furthermore, a gradient-based technique (temporal sensitivity analysis) was introduced in order to identify the more relevant and more class-specific features. Overall, the proposed CNN, tested on motor execution and motor imagery EEG signals, outperformed other state-of-the-art CNNs and a traditional machine learning algorithm. The analyses on the design choices and training strategies confirmed that the proposed architecture is a good compromise between decoding performance and an efficient use of trainable parameters. The post-hoc interpretation techniques suggest that the features learned by the convolutional module matched well-known EEG motor-related activity, both in the frequency and spatial domains. While Sinc-ShallowNet was applied only to motor-related EEG decoding, it was not specifically tailored to decoding sensorimotor rhythm and may be used also in other EEG decoding tasks (e.g., P300 detection or other ERP classification tasks). Furthermore, if a specific decoding task benefits from deeper architectures, the interpretable and optimized convolutional module proposed in Sinc-ShallowNet could be easily employed to design deeper CNNs by stacking more convolutional layers on it. In particular, due to its augmented interpretability, Sinc-ShallowNet or a deeper CNN based on it, may be applied to investigate cognitive and/or motor aspects for which the distinctive EEG correlates are less known (e.g., attention, emotion, creativity, movement trajectory/kinematics etc.).

5.6. SUPPLEMENTARY MATERIALS

5.6.1. State-of-the-art CNNs

The SOA CNN architectures considered for the comparison with Sinc-ShallowNet are reported in Supplementary Tables 5.1, 5.2 and 5.3, respectively for EEGNet [11], DeepConvNet and ShallowNet [14].

Supplementary Table 5.1 – Architecture details of EEGNet. Each layer is provided with its name, main hyper-parameters, number of trainable parameters and activation function. See Section 5.2.3 for the meaning of the symbols. *Kernel maximum norm constraint at 1 and 0.25, respectively for the depthwise convolutional and fully-connected layers.

Layer name	Hyper-parameters	Number of parameters	Activation
Input		0	
Conv2D	$K_1 = 8$ $F_1 = (1,65)$ $S_1 = (1,1)$ $P_1 = (0,32)$	$F_1[1] \cdot K_1$	Linear
BatchNorm2D	$m = 0.99$	$2 \cdot K_1$	
DW-Conv2D*	$K_2 = K_1 \cdot D_2$ $F_2 = (C, 1)$ $D_2 = 2$ $S_2 = (1,1)$ $P_2 = (0,0)$	$F_2[0] \cdot K_2$	Linear
BatchNorm2D	$m = 0.99$	$2 \cdot K_2$	
Activation	$\alpha = 1$	0	ELU
AvgPool2D	$F_{p1} = (1,8)$ $S_{p1} = (1,8)$	0	
Dropout	$p = 0.5$	0	
Sep-Conv2D	$K_3 = K_2 \cdot D_3$ $F_3 = (1,33)$ $D_3 = 1$ $S_3 = (1,1)$ $P_3 = (0,16)$	$F_3[1] \cdot K_3 + (K_3)^2$	Linear
BatchNorm2D	$m = 0.99$	$2 \cdot K_3$	
Activation	$\alpha = 1$	0	ELU
AvgPool2D	$F_{p2} = (1,16)$ $S_{p2} = (1,16)$	0	
Dropout	$p = 0.5$	0	
Flatten		0	
Fully-Connected*	$N_c = 4$	$N_c \cdot T_{p2} \cdot K_3 + N_c$	
Activation		0	Softmax

Supplementary Table 5.2 – Architecture details of DeepConvNet. Each layer is provided with its name, main hyper-parameters, number of trainable parameters and activation function. See Section 5.2.3 for the meaning of the symbols. *Kernel maximum norm constraint at 2 and 0.5, respectively for the convolutional and fully-connected layers. For numerical stability, batch normalization ε parameter was set to 1e-5.

Layer name	Hyper-parameters	Number of parameters	Activation
Input		0	
Conv2D*	$K_1 = 25$ $F_1 = (1,10)$ $S_1 = (1,1)$ $P_1 = (0,0)$	$F_1[1] \cdot K_1 + K_1$	Linear
Conv2D*	$K_2 = 25$ $F_2 = (C, 1)$ $S_2 = (1,1)$ $P_2 = (0,0)$	$K_1 \cdot F_2[0] \cdot K_2$	Linear
BatchNorm2D	$m = 0.9$	$2 \cdot K_2$	
Activation	$\alpha = 1$	0	ELU
MaxPool2D	$F_{p1} = (1,2)$ $S_{p1} = (1,2)$	0	
Dropout	$p = 0.5$	0	
Conv2D*	$K_3 = 50$ $F_3 = (1,10)$ $S_3 = (1,1)$ $P_3 = (0,0)$	$K_2 \cdot F_3[1] \cdot K_3$	Linear
BatchNorm2D	$m = 0.9$	$2 \cdot K_3$	
Activation	$\alpha = 1$	0	ELU
MaxPool2D	$F_{p2} = (1,2)$ $S_{p2} = (1,2)$	0	
Dropout	$p = 0.5$	0	
Conv2D*	$K_4 = 100$ $F_4 = (1,10)$ $S_4 = (1,1)$ $P_4 = (0,0)$	$K_3 \cdot F_4[1] \cdot K_4$	Linear
BatchNorm2D	$m = 0.9$	$2 \cdot K_4$	
Activation	$\alpha = 1$	0	ELU
MaxPool2D	$F_{p3} = (1,2)$ $S_{p3} = (1,2)$	0	
Dropout	$p = 0.5$	0	
Conv2D*	$K_5 = 200$ $F_5 = (1,10)$ $S_5 = (1,1)$ $P_5 = (0,0)$	$K_4 \cdot F_5[1] \cdot K_5$	Linear
BatchNorm2D	$m = 0.9$	$2 \cdot K_5$	
Activation	$\alpha = 1$	0	ELU
MaxPool2D	$F_{p4} = (1,2)$ $S_{p4} = (1,2)$	0	
Flatten		0	
Fully-Connected*	$N_c = 4$	$N_c \cdot T_{p4} \cdot K_5 + N_c$	
Activation		0	Softmax

Supplementary Table 5.3 – Architecture details of ShallowNet. Each layer is provided with its name, main hyper-parameters, number of trainable parameters and activation function. See Section 5.2.3 for the meaning of the symbols. *Kernel maximum norm constraint at 2 and 0.5, respectively for the convolutional and fully-connected layers. For numerical stability, batch normalization ε parameter was set to 1e-5, while the log function input was clipped at $\varepsilon = 1e-6$.

Layer name	Hyper-parameters	Number of parameters	Activation
Input		0	
Conv2D*	$K_1 = 40$ $F_1 = (1,25)$ $S_1 = (1,1)$ $P_1 = (0,0)$	$F_1[1] \cdot K_1 + K_1$	Linear
Conv2D*	$K_2 = 40$ $F_2 = (C,1)$ $S_2 = (1,1)$ $P_2 = (0,0)$	$K_1 \cdot F_2[0] \cdot K_2$	Linear
BatchNorm2D	$m = 0.9$	$2 \cdot K_2$	
Activation	$\alpha = 1$	0	Square
AvgPool2D	$F_p = (1,75)$ $S_p = (1,15)$	0	
Activation		0	Log
Dropout	$p = 0.5$	0	
Flatten		0	
Fully-Connected*	$N_c = 4$	$N_c \cdot T_p \cdot K_2 + N_c$	
Activation		0	Softmax

5.6.2. FBCSP+rLDA

As traditional machine learning decoding algorithm, we used a pipeline previously validated and adopted in Schirrmeister et al. [14]. Two different overlapped filter banks were designed for ME and MI-EEG signals. Starting from a frequency value of 4 Hz, frequency bands were selected with 6 Hz width and overlap factor of 3 Hz up to 16 Hz, and frequency bands with 8 Hz width and overlap factor of 4 Hz for frequencies above 13 Hz (up to 121 Hz and 37 Hz for ME- and MI-EEG signals, respectively). Thus, 29 and 8 band-pass filters were computed for ME- and MI-EEG signals. For each of these manually designed filters, EEG signals were band-pass filtered. Two CSP filter pairs (four filters total) for each filter bank were computed on the training data. Since a few spatial filters computed often are enough to reach good decoding performance while using all the spatial filters may lead to overfitting [48,49], we included the feature selection procedure adopted in [14].

As the decoding task is multi-class, the problem was transformed into several binary classification tasks via a one-vs-one reduction (OVO), where binary classifiers learned to discriminate each pair of classes. Then, a majority weighted voting was applied at prediction time. To do so, we trained a rLDA classifier with shrinkage regularization [50], widely used in EEG decoding [3] for each pair of classes, summed up the classifier outputs and the class with higher sum was decoded as the predicted one [49].

Comparing FBCSP+rLDA results – obtained by re-implementing the steps adopted in [14] – with another study [20] that used the same MI dataset, no significant difference was observed

($P = 0.441$ Wilcoxon signed-ranked test, average accuracy across subjects: 67.5 vs. 67.0 % [20]). This validated the FBCSP+rLDA re-implementation adopted in this study.

5.7. REFERENCES

[1] McFarland D J, Anderson C W, Muller K-, Schlogl A and Krusienski D J 2006 BCI meeting 2005-workshop on BCI signal processing: feature extraction and translation *IEEE Transactions on Neural Systems and Rehabilitation Engineering* **14** 135–8

[2] Bashashati A, Fatourechi M, Ward R K and Birch G E 2007 A survey of signal processing algorithms in brain–computer interfaces based on electrical brain signals *Journal of Neural Engineering* **4** R32–57

[3] Lotte F, Bougrain L, Cichocki A, Clerc M, Congedo M, Rakotomamonjy A and Yger F 2018 A review of classification algorithms for EEG-based brain–computer interfaces: a 10 year update *Journal of Neural Engineering* **15** 031005

[4] Mak J N and Wolpaw J R 2009 Clinical Applications of Brain-Computer Interfaces: Current State and Future Prospects *IEEE Reviews in Biomedical Engineering* **2** 187–99

[5] Ang K K, Chin Z Y, Wang C, Guan C and Zhang H 2012 Filter Bank Common Spatial Pattern Algorithm on BCI Competition IV Datasets 2a and 2b *Frontiers in Neuroscience* **6** 39

[6] Roy Y, Banville H, Albuquerque I, Gramfort A, Falk T H and Faubert J 2019 Deep learning-based electroencephalography analysis: a systematic review *Journal of Neural Engineering* **16** 051001

[7] Borra D, Fantozzi S and Magosso E 2020 Convolutional Neural Network for a P300 Brain-Computer Interface to Improve Social Attention in Autistic Spectrum Disorder *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1837–43

[8] Borra D, Fantozzi S and Magosso E 2020 EEG Motor Execution Decoding via Interpretable Sinc-Convolutional Neural Networks *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1113–22

[9] Cecotti H and Graser A 2011 Convolutional Neural Networks for P300 Detection with Application to Brain-Computer Interfaces *IEEE Transactions on Pattern Analysis and Machine Intelligence* **33** 433–45

[10] Farahat A, Reichert C, Sweeney-Reed C and Hinrichs H 2019 Convolutional neural networks for decoding of covert attention focus and saliency maps for EEG feature visualization *Journal of Neural Engineering*

[11] Lawhern V J, Solon A J, Waytowich N R, Gordon S M, Hung C P and Lance B J 2018 EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces *Journal of Neural Engineering* **15** 056013

[12] Leeuwen K G van, Sun H, Tabaeizadeh M, Struck A F, Putten M J A M van and Westover M B 2019 Detecting abnormal electroencephalograms using deep convolutional networks *Clinical Neurophysiology* **130** 77–84

[13] Manor R and Geva A B 2015 Convolutional Neural Network for Multi-Category Rapid Serial Visual Presentation BCI *Frontiers in Computational Neuroscience* **9** 146

[14] Schirrmeister R T, Springenberg J T, Fiederer L D J, Glasstetter M, Eggensperger K, Tangermann M, Hutter F, Burgard W and Ball T 2017 Deep learning with convolutional neural networks for EEG decoding and visualization *Human brain mapping* **38** 5391–420

[15] Shamwell J, Lee H, Kwon H, Marathe A R, Lawhern V and Nothwang W 2016 Single-trial EEG RSVP classification using convolutional neural networks *Micro- and Nanotechnology Sensors, Systems, and Applications VIII* vol 9836, ed T George, A K Dutta and M S Islam (SPIE) pp 373–82

[16] Tang Z, Li C and Sun S 2017 Single-trial EEG classification of motor imagery using deep convolutional neural networks *Optik* **130** 11–8

[17] Zeng H, Wu Z, Zhang J, Yang C, Zhang H, Dai G and Kong W 2019 EEG Emotion Classification Using an Improved SincNet-Based Deep Learning Model *Brain Sciences* **9**
[18] Zhao D, Tang F, Si B and Feng X 2019 Learning joint space–time–frequency features for EEG decoding on small labeled data *Neural Networks* **114** 67–77
[19] Bashivan P, Rish I, Yeasin M and Codella N 2015 Learning Representations from EEG with Deep Recurrent-Convolutional Neural Networks *CoRR* **abs/1511.06448**
[20] Sakhavi S, Guan C and Yan S 2015 Parallel convolutional-linear neural network for motor imagery classification *2015 23rd European Signal Processing Conference (EUSIPCO)* pp 2736–40
[21] Tabar Y R and Halici U 2016 A novel deep learning approach for classification of EEG motor imagery signals *Journal of Neural Engineering* **14** 016003
[22] Montavon G, Samek W and Müller K-R 2018 Methods for interpreting and understanding deep neural networks *Digital Signal Processing* **73** 1–15
[23] Jonas S, Rossetti A O, Oddo M, Jenni S, Favaro P and Zubler F 2019 EEG-based outcome prediction after cardiac arrest with convolutional neural networks: Performance and visualization of discriminative features *Human Brain Mapping* **40** 4606–17
[24] Ravanelli M and Bengio Y 2018 Speaker Recognition from Raw Waveform with SincNet *2018 IEEE Spoken Language Technology Workshop (SLT)* pp 1021–8
[25] Simonyan K, Vedaldi A and Zisserman A 2014 Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps *arXiv:1312.6034 [cs]*
[26] Paszke A, Gross S, Chintala S, Chanan G, Yang E, DeVito Z, Lin Z, Desmaison A, Antiga L and Lerer A 2017 Automatic differentiation in PyTorch *NIPS-W*
[27] Ball T, Demandt E, Mutschler I, Neitzel E, Mehring C, Vogt K, Aertsen A and Schulze-Bonhage A 2008 Movement related activity in the high gamma range of the human EEG *NeuroImage* **41** 302–10
[28] Crone N E, Miglioretti D L, Gordon B and Lesser R P 1998 Functional mapping of human sensorimotor cortex with electrocorticographic spectral analysis. II. Event-related synchronization in the gamma band. *Brain : a journal of neurology* **121 (Pt 12)** 2301–15
[29] Pfurtscheller G 1981 Central beta rhythm during sensorimotor activities in man *Electroencephalography and Clinical Neurophysiology* **51** 253–64
[30] Pfurtscheller G and Aranibar A 1977 Event-related cortical desynchronization detected by power measurements of scalp EEG *Electroencephalography and Clinical Neurophysiology* **42** 817–26
[31] Pfurtscheller G and Berghold A 1989 Patterns of cortical activation during planning of voluntary movement *Electroencephalography and Clinical Neurophysiology* **72** 250–8
[32] Pfurtscheller G, Brunner C, Schlögl A and Silva F H L da 2006 Mu rhythm (de)synchronization and EEG single-trial classification of different motor imagery tasks *NeuroImage* **31** 153–9
[33] Pfurtscheller G and Silva F H L da 1999 Event-related EEG/MEG synchronization and desynchronization: basic principles *Clinical Neurophysiology* **110** 1842–57
[34] Pfurtscheller G, Flotzinger D and Neuper C 1994 Differentiation between finger, toe and tongue movement in man based on 40 Hz EEG *Electroencephalography and Clinical Neurophysiology* **90** 456–60
[35] Tangermann M, Müller K-R, Aertsen A, Birbaumer N, Braun C, Brunner C, Leeb R, Mehring C, Miller K J, Mueller-Putz G, and others 2012 Review of the BCI competition IV *Frontiers in neuroscience* **6** 55
[36] Lotte F 2015 Signal Processing Approaches to Minimize or Suppress Calibration Time in Oscillatory Activity-Based Brain–Computer Interfaces *Proceedings of the IEEE* **103** 871–90

[37] Chollet F 2016 Xception: Deep Learning with Depthwise Separable Convolutions *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 1800–7
[38] Ioffe S and Szegedy C 2015 Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift *Proceedings of the 32nd International Conference on Machine Learning* Proceedings of Machine Learning Research vol 37, ed F Bach and D Blei (Lille, France: PMLR) pp 448–56
[39] Clevert D-A, Unterthiner T and Hochreiter S 2015 Fast and accurate deep network learning by exponential linear units (elus) *arXiv preprint*
[40] Srivastava N, Hinton G, Krizhevsky A, Sutskever I and Salakhutdinov R 2014 Dropout: a simple way to prevent neural networks from overfitting *The Journal of Machine Learning Research* **15** 1929–58
[41] Glorot X and Bengio Y 2010 Understanding the difficulty of training deep feedforward neural networks *Proceedings of the thirteenth international conference on artificial intelligence and statistics* pp 249–56
[42] Kingma D P and Ba J 2017 Adam: A Method for Stochastic Optimization *arXiv:1412.6980 [cs]*
[43] Goodfellow I J, Warde-Farley D, Mirza M, Courville A and Bengio Y 2013 Maxout networks *Proceedings of the 30th International Conference on International Conference on Machine Learning-Volume 28* (JMLR. org) p III–1319
[44] Liu M, Wu W, Gu Z, Yu Z, Qi F and Li Y 2018 Deep learning based on Batch Normalization for P300 signal detection *Neurocomputing* **275** 288–97
[45] Ang K K, Chin Z Y, Zhang H and Guan C 2008 Filter bank common spatial pattern (FBCSP) in brain-computer interface *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)* (IEEE) pp 2390–7
[46] Benjamini Y and Hochberg Y 1995 Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing *Journal of the Royal Statistical Society. Series B (Methodological)* **57** 289–300
[47] Mirnaziri M, Rahimi M, Alavikakhaki S and Ebrahimpour R 2013 Using Combination of μ,β and γ Bands in Classification of EEG Signals *Basic and clinical neuroscience*
[48] Blankertz B, Tomioka R, Lemm S, Kawanabe M and Muller K 2008 Optimizing Spatial filters for Robust EEG Single-Trial Analysis *IEEE Signal Processing Magazine* **25** 41–56
[49] Chin Z Y, Ang K K, Wang C, Guan C and Zhang H 2009 Multi-class filter bank common spatial pattern for four-class motor imagery BCI *2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society* pp 571–4
[50] Ledoit O and Wolf M 2004 A well-conditioned estimator for large-dimensional covariance matrices *Journal of Multivariate Analysis* **88** 365–411

CHAPTER 6: DESIGN OF AN INTERPRETABLE AND MULTI-SCALE CNN FOR MOTOR DECODING AND ANALYSIS IN THE FREQUENCY AND SPATIAL DOMAINS

The study reported in this chapter refers to the submitted journal paper entitled "An interpretable multi-scale convolutional neural network for EEG motor trajectory decoding and analysis of kinematic neural signatures" D. Borra, V. Mondini, E. Magosso, G. R. Müller-Putz. Submitted to *IEEE Transactions on Neural Networks and Learning Systems*. In this study, the interpretable CNN presented in Chapter 5 was modified by exploiting the promising results obtained for P300 decoding in Chapter 4. In particular, the multi-scale structure of deeper layers operating in the temporal domain (see Chapter 4) was included in the interpretable CNN structure used in Chapter 5, thus obtaining a novel interpretable and multi-scale CNN for motor decoding. Notably, the motor decoding problem faced here was more difficult than that in Chapter 5 since it required the prediction of 2D position and velocity trajectories, instead of a simpler categorical variable. Different training strategies were used to evaluate the interpretable multi-scale CNN. Lastly, an intermediate explanation technique was developed to investigate the neural signatures related to position and velocity in the frequency and spatial domains.

Continuous decoding of voluntary movement has been recently explored to provide natural control in EEG-based Brain-computer interfaces (BCIs). This is typically performed on low-frequency (< 3 Hz) electroencephalogram (EEG) with linear regression techniques such as partial least square regression, which may be combined with unscented Kalman filters (PLS+UKF) to better reconstruct the amplitude of the trajectories. Convolutional neural networks (CNNs), by exploiting automatic feature learning from slightly pre-processed signals, are emerging as powerful algorithms to decode information from neural time series, and to highlight the neural signatures related to the decoded brain states. In this study, we designed a light, multi-scale and interpretable CNN (ICNN) to decode upper-limb 2-D positions and velocities in a pursuit tracking task performed by 13 participants, leaving the ICNN free to explore also high frequencies of the EEG. The network incorporates interpretable components in its design to ease the interpretation of the spectral and spatial features. A data-driven analysis framework based on the ICNN was designed to highlight the most relevant spectral and spatial EEG signatures encoding position and velocity. The ICNN better reconstructed the trajectory amplitudes than the PLS+UKF, while performing on par or even outperforming state-of-the-art CNNs, and providing at the same time a more interpretable spectral and spatial feature learning. The ICNN was evaluated under different training conditions, including within-subject, cross-subject, and transfer learning, to simulate its application in different BCI scenarios. Transfer learning improved the performance when using few training trials, which may enable in the future a reduction of BCI calibration times. The ICNN-based analysis framework highlighted the highest relevance of delta range for kinematics, and of higher frequencies, namely, alpha, beta, low-gamma, for a cluster of subjects. Furthermore, contralateral central to parieto-occipital sites were the most relevant, reflecting the involvement of both sensorimotor, visual processing, and eye-hand movement coordination areas.

6.1. INTRODUCTION

Recent efforts in Brain-Computer Interfaces (BCIs) research have been focusing on the fine reconstruction of voluntary movement trajectories from brain signals, so to be able to control an actuator (e.g., a robotic arm, or a neuroprosthesis) in a more intuitive and natural way [1–5]. Voluntary movement trajectories have been decoded from invasively recorded signals, like electrocorticographic [6,7] or intracortical recordings [5,8], and from non-invasive recordings as well, like magnetoencephalographic [9–13] and electroencephalographic (EEG) recordings [13–20]. More recently, EEG-based trajectory decoding was also employed for closed-loop control [21,22]. Trajectory decoding from the EEG mainly exploits the low-frequency (LF) range of the signal, i.e., < 3 Hz [23], and uses low-pass filtered EEG signals as input to decoders [24] to predict the end-effector positions and velocities [16]. Recent research, however, suggests that higher frequency components (e.g., beta and low-gamma) may carry additional information on the movement [17,20] and, therefore, using only LF-EEG as input might negatively affect trajectory decoding.

The state-of-the-art (SOA) widely adopts linear models for trajectory decoding, such as partial least squares (PLS) regression [15,21,25], or the combination of PLS with Kalman filters (KF), i.e., PLS+KF, to integrate the information of different decoding models [21]. However, when the only information of directional parameters (e.g., positions and velocities) was used for the decoding, an amplitude mismatch between the decoded and the actual trajectories could be observed [15,21], thus, suggesting a role of non-directional parameters (e.g., distance and speed) in reconstructing the amplitude [22], [13]. To integrate both types of information, a new PLS + Unscented Kalman Filter (PLS+UKF) decoder was introduced [6,13,22]. The PLS+UKF model was successful in alleviating the amplitude mismatch, and used both offline [13] and online [22] to decode the movement from the LF-EEG.

Overall, these decoders have been mainly trained separately for each subject (within-subject), due to inter-subject EEG variability, while the possibility to transfer the knowledge from other subjects to a new one (transfer learning, TL), a practice useful to reduce BCI calibration times, has not been explored.

Deep Learning (DL) algorithms, in particular Convolutional Neural Networks (CNNs), have been recently applied to EEG decoding in several domains [26,27] such as emotion classifications, classification of executed or imagined movements (typically decoding the body part involved in the movement, e.g., hand vs. feet), and event-related potential detection (e.g., P300). In general, deep neural networks consist in the sequence of layers of artificial neurons and, depending on the established connections between neurons, feed-forward or recurrent neural networks can be designed. CNNs are feed-forward neural networks that perform convolution at least in one layer within the network. CNNs learn hierarchically structured features [28] from the input EEG and, in contrast to traditional machine learning approaches, are able to automatically learn the most relevant features from raw or slightly pre-processed signals, without discarding a priori components or features of the input EEG. CNNs have been proven to outperform traditional machine learning decoders in EEG decoding tasks, including motor classification [29–40]. In addition, CNN-based EEG decoders have been evaluated under different training strategies, including within-subject, cross-subject and TL, thus promoting the use of CNNs in practice for BCIs [33,35,41]. Across studies, the CNN

architectures adopted for EEG motor classification share some common elements in their structure. Typically, CNNs first learn features in the temporal and then in the spatial domains, by separately performing convolutions in the time and space domains of the multi-variate EEG input [29,30,32,36,39]. Then, more abstract features in the temporal domain are learned in subsequent deeper layers. One recent solution to learn deep temporal features better capturing intra- and inter-subject variability, consist in learning features at multiple time scales, realizing multi-scale CNNs; this solution resulted beneficial compared to learning features at a single time scale in EEG classification tasks, including motor [37–39] and P300 [33,35] classification.

Despite the promising results of CNN-based EEG decoding, CNNs suffer of some drawbacks. i) They introduce many trainable parameters (even >100K [29]), i.e., parameters to fit during training, leading to decoders prone to overfit small datasets. ii) They introduce many hyper-parameters, i.e., parameters that define the functional form of the CNN (e.g., number of convolutional filters) and, thus, the optimal structure of the network (in terms of performance) is not known a priori. iii) They lack in interpretability of the learned features.

In the context of EEG decoding, efforts have been made to overcome these limitations. In particular, lightweight CNNs have been proposed, adopting specialized light convolutions (i.e., that introduce less trainable parameters) [30,32,33], such as depthwise and separable convolutions [42]. Furthermore, sensitivity analyses on hyper-parameters have been performed to examine the role of each hyper-parameter in terms of network performance [29,32,33]. Lastly, recent methodological advancements have been proposed to increase the interpretability of the learned features and to exploit the full potentialities of the automatic feature learning performed by the CNN. To this aim, interpretability has been directly incorporated into the CNN structure by designing *interpretable* layers, i.e., layers whose trainable parameters are directly interpretable in a given domain (e.g., frequency domain), realizing an *interpretable* CNN (ICNN) [32,40]. In addition to interpretable layers, *explanation techniques* (ETs) such as saliency representations [43], reporting the gradient of a target output (e.g., a motor state) with respect to each input sample, have been used to explain the CNN decision [32,35,41,44], highlighting the EEG features in the spatial, temporal or spectral domains that resulted most discriminative for the specific output. Therefore, by properly designing an ICNN and combining it with an ET, a data-driven non-linear analysis tool (ICNN+ET) can be realized to study EEG signatures associated to the decoded outputs.

All the previous approaches have been applied to different EEG classification problems, including movement classification. However, their application to the continuous regression of movement kinematics from EEG remains unexplored. This represents an important limit; indeed, not only CNNs could increase the performance of EEG-based trajectory decoding by exploiting also high-frequency components of the input, but also the automatic feature learning performed by CNN may be employed to investigate the EEG signatures related to kinematics in a data-driven way.

In this study, we aim at contributing to EEG trajectory decoding by using a non-linear decoder based on an ICNN. The addressed decoding problem is the continuous reconstruction of the upper-limb 2-D velocities and positions in a pursuit tracking task. In particular, the following issues were addressed:

i. Perform trajectory decoding from the EEG without focusing only on LF components but leaving the learning system the capability to freely explore also higher frequency

components of the input EEG that may result relevant for the decoding.

ii. Design a light and interpretable multi-scale CNN to continuously predict position and velocity from the EEG; this was obtained by integrating inside the CNN an interpretable spectral and spatial feature extractor, previously validated for motor classification [32], and a light temporal feature extractor that learns temporal patterns at two different time scales in parallel. This approach was compared with SOA decoders, including a traditional non-linear PLS+UKF algorithm [13] specifically proposed for trajectory decoding, and decoders based on CNNs (DeepConvNet and ShallowConvNet) [29] that were previously proposed for motor classification.

iii. Study the decoder using different training strategies, including within-subject, leave-one-subject-out (i.e., cross-subject) and transfer learning, to explore the potentialities of CNNs to transfer the knowledge from previously recorded users on a new user approaching the BCI, to reduce calibration time on the new user.

iv. Design an ICNN+ET algorithm devoted to highlight - in a data-driven way - the most relevant EEG signatures encoding positions and velocities in the frequency and spatial domains. That is, in this study we focused on the design of a framework for decoding kinematics from the EEG and for analyzing kinematic-related EEG signatures.

6.2. MATERIALS AND METHODS

6.2.1. Single-trial EEG trajectory decoding via CNNs

Let's assume that from each subject, EEG signals and a set of variables to be predicted were continuously recorded for several trials. Then, the dataset $D^{(s)}$ associated to the s-th subject can be expressed as:

$$D^{(s)} = \left\{\left(X_0^{(s)}, Y_0^{(s)}\right), \dots, \left(X_i^{(s)}, Y_i^{(s)}\right), \dots, \left(X_{M^{(s)}-1}^{(s)}, Y_{M^{(s)}-1}^{(s)}\right)\right\}, \quad (6.1)$$

where $X_i^{(s)} \in \mathbb{R}^{C\times T}$ $(0 \le i \le M^{(s)} - 1)$ contains the pre-processed EEG signals of the i-th trial recorded from the C electrode sites and consisting of T time samples, while $Y_i^{(s)} \in \mathbb{R}^{K\times T}$ contains the K pre-processed time series to be predicted organized by rows, recorded for T time samples (same as EEG). When performing 2-D kinematic decoding, these variables could correspond to the 2-D position and/or 2-D velocity components of the hand (see Section 6.2.2), resulting in a continuous decoding of kinematics variables from single-trial EEG. To perform such decoding, the learning system predicts the kinematic variables at each time sample by using a buffer of EEG signals, hereafter denoted as "chunk", and consisting of T_z time samples sampled from the original multi-variate time series. By indicating with T_s the stride used to sample these chunks (see Section 6.6.1 of Supplementary Materials), we can write:

$$\begin{cases} Z_{i,j}^{(s)} = X_i^{(s)}[:, jT_s : jT_s + T_z - 1] \in \mathbb{R}^{C\times T_z} \\ y_{i,j}^{(s)} = Y_i^{(s)}[jT_s + T_z - 1] \in \mathbb{R}^{K} \end{cases}, 0 \le j \le L - 1, \quad (6.2)$$

where L denotes the number of chunks that could be extracted using T_z and T_s as chunk size and stride, respectively, i.e., $L = (T - T_z)/T_s + 1$.

The objective decoding problem can be formalized as the optimization of the parametrized regressor f implemented with a CNN, $f\left(Z_{i,j}^{(s)}; \theta\right): \mathbb{R}^{C\times T_z} \to \mathbb{R}^K$, with its parameters contained in θ and that must be learned from a training set of examples to assign the correct label to unseen examples. $Z_{i,j}^{(s)}$ represents the CNN input, containing a chunk of the multi-variate brain activity organized in a 2D array of shape (C, T_z), with electrodes along the height and time steps along the width. $y_{i,j}^{(s)}$ represents the CNN output, containing the K values of the variables to predict organized in a 1D array of shape $(K, 1)$. See Section 6.2.2 for the definition of C, T_z, K in this application. The dataset $D^{(s)}$ (see Equation 6.1) was divided into a training set used to optimize θ, and a test set used to test the algorithm on unseen examples. In addition, a separate validation set was extracted from the training set to define the stop criterion of the optimization. See Sections 6.2.2 and 6.2.4 for additional details.

6.2.2. Data description and pre-processing

In this study, we reanalyzed the data of the Graz BCI group recorded in [21] and [22], including the EEG signals of 13 healthy subjects (aged of 27±4 years, mean±standard deviation, 7 females, 1 left-handed) as they were performing a pursuit tracking task with their right hand/arm. During the experiment, subjects were asked to track a moving object displayed on a screen using a robotic arm; the latter was controlled by a mixture of hand kinematics and trajectories decoded from the EEG (see Figure 6.1A). The experiment was composed by a

calibration and online feedback phases, both collected in runs (5 calibration runs, 6 feedback runs). Each run was composed of 10 trials in which the object was tracked for 23s. Crucially, the trajectories of the object were generated offline to ensure uncorrelated positions and velocities across and within horizontal (x) and vertical (y) coordinates. Two additional special runs, called "eyeruns", were performed to collect rest data, saccadic eye movements, and blinks, to fit a regression model to attenuate eye movement artifacts [45]. During the calibration phase, the robot was entirely controlled by the hand kinematics. Afterwards, a linear [21] or non-linear [22] decoder could be fitted so to predict the kinematics from the EEG. During the online feedback phase, the subject could then gradually receive feedback on the decoded movements, as the control signal of the robot was progressively switched from hand kinematics to EEG-based decoded trajectories. In this study, we used the signals collected during the calibration phase as training set, and the signals collected during the online feedback phase as test set, as performed in [21,22]. Therefore, the training and test sets were composed by 50 and 60 trials (each one lasting 23 s), respectively.

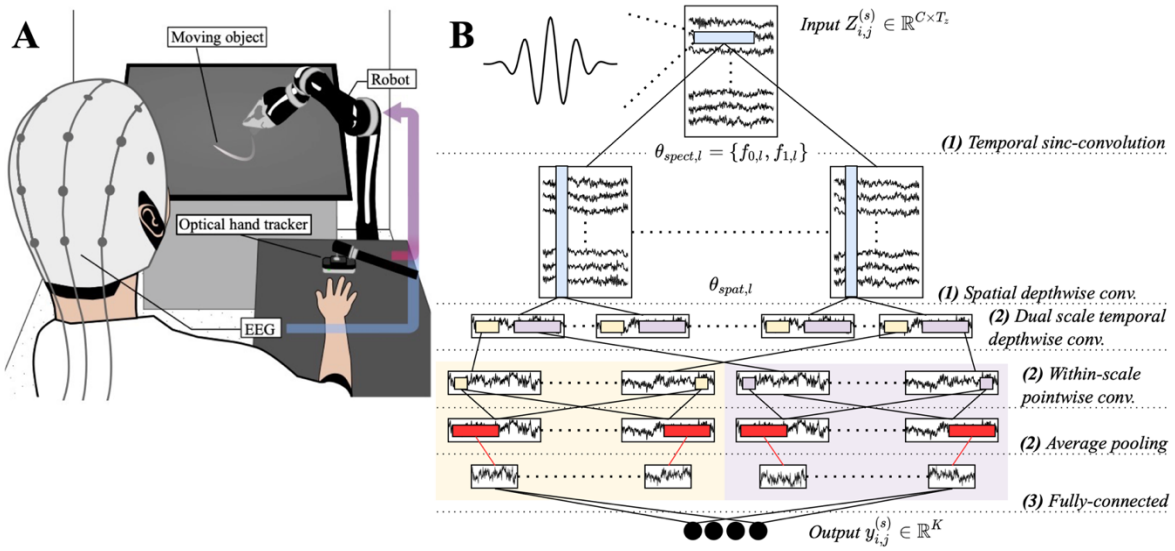


Figure 6.1 – (A) Schematics of the recording setup. (B) MS-Sinc-ShallowNet structure. The input had shape of $(C, T_z) = (53,100)$, while the output of $(K, 1) = (4,1)$. The main layers are listed on the right side, while block IDs are reported within brackets. Boxes represent the output feature maps of each layer, and colored rectangles represent convolutional and pooling (red) kernels.

During the recordings, the 2-D positions and velocities of the right hand ($K = 4$) were recorded using an optical hand tracking module, together with the EEG signals from 64 [21] or 60 [22] electrodes placed on the scalp according to the 10-10 system, of which a common subset of 53 electrodes between the two studies ($C = 53$) was used here to perform trajectory decoding. Reference and ground electrodes were placed at the right mastoid and AFz, respectively. Additional electrodes were placed around the eyes to record the electrooculogram (EOG). Both the EEG and EOG signals were recorded at 500 Hz, while the hand trajectories with a variable sampling rate depending on the speed of the movement. The 2-D kinematic hand trajectories were low-pass filtered using a cut-off frequency of 4 Hz and downsampled at 100 Hz. Regarding the EEG, a pre-processing similar to the ones in [21,22] was used; however, as in this study we were interested in leaving the learning system free to explore more frequency components than in [21,22], the EEG signals were band-pass filtered more broadly.

Specifically, the EEG pre-processing included: 1) Zero-phase, high-pass filtering (1st order Butterworth) with a cut-off frequency of 0.18 Hz; 2) Zero-phase, low-pass filtering (4th order Butterworth) with a cut-off frequency of 40 Hz; 3) Notch filtering at 50 Hz and 100Hz; 4) Downsampling at 100 Hz; 5) Bad channel marking via visual inspection, and linear interpolation from the 4 nearest neighboring channels; 6) Eye artifact correction based on SGEYESUB algorithm [45] (after fitting the algorithm on signals of the eyeruns); 7) Common average referencing; 8) Interpolation of the slow drifts/occasional electrode pops via HEAR algorithm [41] (fitting the algorithm on eye-corrected signals of the eyeruns).

Subsequently, the signals were subjected to the processing reported in Equation 6.2, specifically by using $T_z = 100$ (i.e., buffer of 1 s), and $T_s = 10$ (i.e., stride of 0.1 s) for EEG trials belonging to the training set, while $T_s = 1$ (i.e., stride of 0.01 s) for the ones belonging to the test set. This was performed to provide an inference on the test examples at the same sampling rate as the kinematics (i.e., 100 Hz), at the same time keeping limited the training time by using a higher stride for the training examples. That is, the number of training EEG chunks per trial resulted $L = (2300 - 100)/10 + 1 = 201$.

6.2.3. The interpretable CNN for trajectory decoding: MS-Sinc-ShallowNet

The ICNN developed in this study, named MS-Sinc-ShallowNet, is a modified version of an ICNN (Sinc-ShallowNet [32]) that we recently proposed for motor classification. Specifically, MS-Sinc-ShallowNet exploits the interpretable spectral and spatial feature extractor of Sinc-ShallowNet [32], placed (at variance with the latter) on top of a light multi-scale (MS) temporal feature extractor. Overall, MS-Sinc-ShallowNet realizes an interpretable architecture that processes the input time series at multiple time scales. The adoption of the fully interpretable spectral and spatial feature extractor allows an easier interpretation of features useful to decode kinematics in the frequency and spatial domains, at the same time keeping limited the model size (defined by the number of trainable parameters). In addition, the adoption of a multi-scale temporal feature learning enables to learn relevant temporal patterns within different time scales simultaneously, without focusing only on one single time scale, a strategy that was already found beneficial while decoding motor [37–39] and cognitive [33,35] states from the EEG.

MS-Sinc-ShallowNet (see Figure 6.1B) was composed by three main blocks, each defined by the sequence layers. Supplementary Table 6.1 reports additional details about the network. Here, the fundamental blocks are described.

i. Interpretable spectral and spatial (ISS) feature extractor

The first block was based on the first layers of Sinc-ShallowNet [32] and was devoted to separately learn spectral and spatial features from the input EEG chunk in an easy interpretable way. The very first layer of the ISS block was a temporal sinc-convolutional layer [32,46,47], learning ${K_0}^{ISS} = 16$ filters with filter size ${F_0}^{ISS} = (1,51)$, unitary stride and zero-padding to preserve the number of input temporal samples. This temporal convolutional layer is devoted to filter each electrode signal in time. Thanks to the use of a sinc-convolutional layer to perform such processing step instead of a conventional convolutional layer, each convolutional filter can be forced to describe a band-pass filter in the temporal domain. Denoting with k_l the l-th

convolutional kernel, in a conventional convolutional layer each filter value (i.e., $k_l[0,n], n \in [0,50]$) has to be learned during the optimization process; conversely, in a sinc-convolutional layer, each filter value is defined by a parametrized function, forcing the overall filter distribution to belong to a specific subset of temporal filters (here only band-pass filters). Therefore, in a sinc-convolutional layer a re-parametrization of each kernel occurs:

$$k_l'[0,n;\{f_{0,l},f_{1,l}\}] = 2f_{1,l}sinc(2\pi f_{1,l}n) - 2f_{0,l}sinc(2\pi f_{0,l}n), 0 \le l \le {K_0}^{ISS} - 1. \quad (6.3)$$

In Equation 6.3, $\{f_{0,l}, f_{1,l}\}$ is the set of trainable parameters related to the l-th kernel, including only the inferior ($f_{0,l}$) and superior ($f_{1,l}$) cutoff frequencies of the band-pass filter. In this way, for each temporal filter the number of trainable parameters reduces from 51 (= ${F_0}^{ISS}[0] \cdot {F_0}^{ISS}[1]$) to 2. Lastly, to alleviate the effects of the inevitable truncation of k_l' on the characteristics of each filter, the multiplication by a Humming window is performed:

$$\begin{cases} k_{w,l}'[0,n;\{f_{0,l},f_{1,l}\}] = k_l'[0,n;\{f_{0,l},f_{1,l}\}] \cdot w[n] \\ w[n] = 0.54 - 0.46\cos\left(\frac{2\pi n}{{F_0}^{ISS}[1]-1}\right) \end{cases}. \quad (6.4)$$

Accordingly, the temporal sinc-convolution computes the convolution between the input and $k_{w,l}'[0,n;\{f_{0,l},f_{1,l}\}]$, learning only the following 2 parameters for each kernel:

$$\theta_{spect,l} = \{f_{0,l}, f_{1,l}\} \in \theta,\ 0 \le l \le {K_0}^{ISS} - 1. \quad (6.5)$$

Thus, the output of this first layer consists of stacked feature maps containing band-pass filtered versions of the input EEG chunk within specific frequency ranges that were explicitly learned during training.

Downstream the temporal sinc-convolutional layer, a spatial depthwise convolutional layer was introduced: for each band-pass filtered map, ${D_1}^{ISS} = 2$ spatial filters were learned having size $(C, 1)$ and unitary stride, i.e., ${D_1}^{ISS}$ spatial combinations of electrodes were learned for each pass-band filtered map (${D_1}^{ISS}$ indicates the depth multiplier). Therefore, a total number of ${K_1}^{ISS} = {K_0}^{ISS} \cdot {D_1}^{ISS} = 32$ spatial filters were learned and constrained to have a norm upper bounded by $c = 1$ (kernel max-norm constraint). This type of convolution does not exploit dense connections across feature maps as in traditional convolutional layers, thus, reducing the number of trainable parameters. In addition, the combination of temporal sinc-convolution with spatial depthwise convolution provides an interpretable spectral-spatial feature learning, as each group of ${D_1}^{ISS}$ spatial filters is strictly tied to a specific band-pass filter, i.e., to a specific frequency range:

$$\theta_{spat,l} = \{\theta_{l0}, \dots \theta_{lk}, \dots, \theta_{l{D_1}^{ISS}-1}\} \in \theta, 0 \le l \le {K_0}^{ISS} - 1, \quad (6.6)$$

indicating with θ_{lk} the k-th spatial filter ($0 \le k \le {D_1}^{ISS} - 1$) tied to the l-th band-pass filter.

This combination enables the design of a fully interpretable spectral-spatial feature extractor, as the parameters of these two first convolutional layers (see Equations 6.5 and 6.6) directly provides the ${K_0}^{ISS}$ pair of cutoff frequencies of the band-pass filters and the associated ${D_1}^{ISS}$ combinations of electrodes exploited to decode the input EEG trial. Hence, the interpretable features are:

$$\begin{cases} \theta_{ISS} = \left\{\theta_{ISS,0}, \dots, \theta_{ISS,l}, \dots, \theta_{ISS,{K_0}^{ISS}-1}\right\} \\ \theta_{ISS,l} = (\theta_{spect,l}, \theta_{spat,l}) \end{cases}. \quad (6.7)$$

Outputs of the ISS feature extractor were activated via an Exponential Linear Unit (ELU)

non-linearity [11], i.e., $f(x) = x, x > 0$ and $f(x) = exp(x) - 1, x \leq 0$, and dropout [49] was applied with dropout rate $p = 0.5$.

ii. Dual-scale temporal (DST) feature extractor

This block was designed to learn temporal features at two time scales from the feature maps provided by the ISS block. Two different and parallel time scales, hereafter called "large" and "short" scales, were used, realizing a sub-network consisting of 2 branches. Separable convolutions were used in each branch to reduce the number of trainable parameters [42], thus, realizing a light dual-scale temporal feature extractor.

At first, each parallel branch included a temporal separable convolutional layer, defined by a temporal depthwise convolution followed by a pointwise convolution. The temporal depthwise convolutional layer learned one temporal pattern per input feature map (i.e., depth multiplier set to 1), unitary stride and zero-padding within each branch. However, it differed in the kernel size ${F_0}^{DST}$ across the two branches, to learn features on different time scales. In particular, ${F_0}^{DST} = (1{,}51)$ and ${F_0}^{DST} = (1{,}25)$, respectively in the large and short scales, corresponding to learning temporal features within windows of approx. 500 and 250 ms. Then, the pointwise convolutional layer learned ${K_1}^{ISS} = 32$ filters of size $(1{,}1)$ with unitary stride, within each branch. This layer optimally recombined the feature maps provided by the depthwise convolution within each scale, separately. That is, at each time scale, one temporal pattern was learned, separately, for each feature map provided by the ISS layer (see Figure 6.1B), and afterwards the optimal combinations of these activations were learned.

Within each branch, the output provided by the temporal separable convolution was activated via ELU non-linearity, and average pooled with pool size and stride of ${F_p}^{DST} = (1{,}10)$ to reduce the number of time steps to be processed in the fully-connected layer of the following block (i.e., reducing from T_z to $T_z//10$, indicating with // the floor division operator). Lastly, dropout [49] was applied with dropout rate $p = 0.5$.

iii. Regressor

This block transforms the feature maps at the output of the DST block into the predicted trajectory values. At first, the feature maps provided by the two parallel branches were concatenated together and reshaped as an array with a single dimension. Then, the flattened feature maps were given as input to a fully-connected layer with $N = K = 4$ units, establishing dense connections with the input feature maps and constraining the weights of these connections to have a norm upper bounded by $c = 1$ (kernel max-norm constraint).

The total number of trainable parameters was 8932 (see Supplementary Table 6.1). The main network hyper-parameters defining the ICNN block 1 and 2 (e.g., the number of band-pass filters, number of spatial filters, and inclusion of batch normalization [50], etc.) and defining the network training (e.g., learning rate) were automatically searched in a preliminary analysis by performing Bayesian optimization [51] (see Section 6.6.2 of Supplementary Materials for further details). The configuration adopted for the ICNN (see Supplementary Table 6.1) was the most frequent configuration across the Bayesian-optimized models. In addition, a sensitivity analysis on the main structural hyper-parameters was conducted (see

Section 6.6.2 of Supplementary Materials), by changing one hyper-parameter at a time and evaluating the performance change compared to the adopted Bayesian-optimized architecture, to understand to what extent hyper-parameters affect the performance, as done in [29,32,33].

6.2.4. Training strategies

In this study, we trained the ICNN with 3 different training strategies, by differently defining the training sets. It is worth noticing that the definition of the test set was the same across training strategies, enabling a fair comparison across them. The training strategies were:

i. Within-subject (WS). Each subject-specific decoder was trained using the subject-specific training set $D^{(s)}$, consisting of the 50 trials of the calibration phase. The ICNNs, besides being trained using all the 50 training trials, were trained by using a progressively increasing number of trials, i.e., from 2 to 10 with a step of 2 trials, randomly sampling the trials to be included in the reduced training set by 10 times. This was performed to simulate practical BCI scenarios where limited training trials are available. Lastly, the test set was defined as the one belonging to the subject the ICNN was trained for, consisting of the 60 trials of the online phase.

ii. Leave-one-subject-out (LOSO). Each decoder was trained using a cross-subject training set. Specifically, for each subject s, named "held-out subject", the training sets of all other subjects were aggregated, i.e., $D^{(-s)} = \{D^{(p)}\}, \forall p \in [0,12], p \neq s$, denoting with "-s" the aggregation across subjects except the held-back one. Therefore, the training set comprised 12*50=600 training trials. Lastly, the test set was defined as the one belonging to the held-out subject (s-th subject). In this way we trained decoders that are cross-subject, because of the training set, and subject-agnostic, as the test set is relative to the subject held out from the training set.

iii. Transfer learning on single subjects (TL-WS). Transfer learning is inspired by the human ability to exploit the knowledge learned in a given domain/task to improve the performance and/or reduce the training time in a different but related domain/task [52]. In this strategy, the knowledge learned on other subjects was transferred to a new subject. As with the WS strategy, subject-specific training sets were used to train subject-specific decoders on the s-th subject, and, thus, the definition of the training and test sets was the same as in the WS strategy. However, differently from the WS strategy in which ICNNs were initialized randomly, in the TL-WS strategy ICNNs were initialized using the trainable parameters obtained during the LOSO strategy when the s-th subject was held-back. Therefore, the knowledge learned during the LOSO strategy, which incorporated inter-subject variability from all other subjects except the held-back one, was transferred on the held-back subject. That is, in this strategy a different initialization is used, potentially representing a better initialization point in the parameter space than the random one, and possibly leading to an improvement in performance and/or to a reduction of the training trials needed to achieve high performance. Therefore, the TL-WS strategy could be useful in a real-life scenario when a new user approaches the BCI and a calibration, as short as possible, is needed to design an accurate decoder.

From the training trials, a validation set was selected by extracting the first 20% portion from each training trial, i.e., by extracting the first 20% of EEG chunks and the corresponding

kinematic values. The ICNN optimization consisted in the minimization of the mean squared error between the predicted and true trajectory values. Adaptive moment estimation (Adam) [53] was used as optimizer with learning rate $lr = 1e - 4$, mini-batch size $bs = 64$, $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for computing the running averages of the gradient and its square, and $\varepsilon = 10^{-8}$ to improve numerical stability. The maximum number of epochs was set to 250 and the training ended when the validation loss did not decrease for 50 consecutive epochs (early stopping). Besides early stopping, MS-Sinc-ShallowNet directly implemented in its structure methodologies devoted to improve generalization, such dropout [49] and kernel max norm constraint.

6.2.5. Interpretation of the spectral and spatial signatures encoding position and velocity

Interpreting the features learned by MS-Sinc-ShallowNet in the WS strategy ($\theta = \theta^{(s)}$, in WS strategy) can provide insights on the most relevant neural signatures of each subject in a data-driven way. The adopted ICNN structure provides interpretable parameters in the array ${\theta_{ISS}}^{(s)}$. As the ICNN processes the input EEG chunks, it filters out motor-unrelated spectral and spatial components while preserving only ones most relevant for the trajectory decoding task. However, these features may have a different importance for the discrimination, meaning that a band-pass filtering in a peculiar frequency range and a subset of electrodes may be relevant to predict position and velocity. Therefore, an explanation technique (ET) was included to highlight the most relevant features (${\theta_{ISS}}^{(s)}$) of each subject for the decoding of positions and velocities. The proposed data-driven EEG analysis consists of the following steps.

Spectral relevance computation

To compute the relevance of each spectral component to predict the positions and velocities, we focused on the ${K_0}^{ISS}$ feature maps from the sinc-convolutional layer. These maps contain the input filtered with the learned band-pass filters. A schematization of the following steps is reported in Figure 6.2.

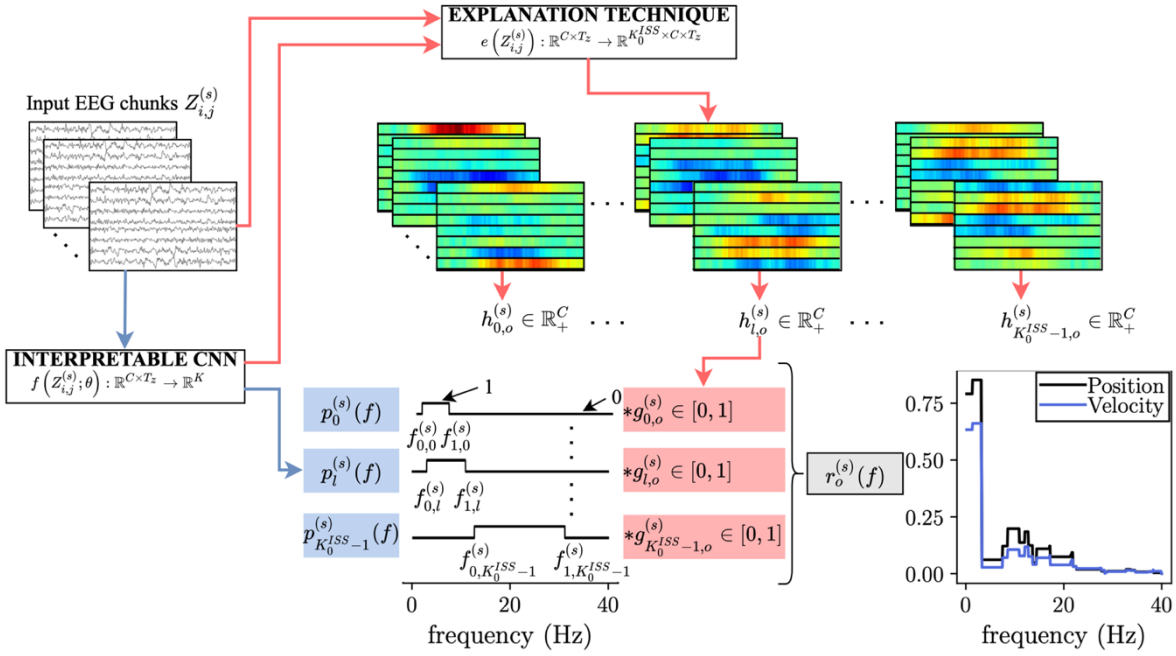


Figure 6.2 – Scheme of the spectral relevance computation. The learned ICNN spectral features $\theta_{spect,l}^{(s)}, 0 \leq l \leq {K_0}^{ISS} - 1$ are extracted and $p_l^{(s)}(f)$ computed (blue boxes and lines). By combining the ICNN with an explanation technique, relevance scores related to each spectral feature are obtained $g_{l,o}^{(s)},\ o \in \{p_x, p_y, v_x, v_y\}$ (red boxes and lines). Then, spectral features and relevance scores are combined to derive the subject-specific spectral relevance of each frequency bin, $r_o^{(s)}(f), o \in \{p, v\}$.

For the EEG chunks of the test set, we evaluated the relevance of each spatio-temporal sample in the feature map to decode the 2D position and velocity components. To do this, we computed saliency maps [43] to quantify, by using gradients, how much a spatio-temporal sample in each filtered input affects the prediction of each kinematic variable (p_x, p_y, v_x, v_y). Therefore, we obtained, for each output variable, one saliency map for each feature map of the first convolutional layer, i.e., $e\left(Z_{i,j}^{(s)}\right)$: $\mathbb{R}^{C \times T_z} \rightarrow \mathbb{R}^{{K_0}^{ISS} \times C \times T_z}$. The so computed saliency maps were averaged across trials $(\forall i)$, chunks $(\forall j)$, and in the temporal domain $(\forall t, 0 \leq t \leq T_z - 1)$. By finally computing the absolute value, the vector quantities $h_{l,o}^{(s)} \in \mathbb{R}_+^C, 0 \leq l \leq {K_0}^{ISS} - 1, o \in \{p_x, p_y, v_x, v_y\}$ can be obtained, with o indicating the output kinematic variable. Finally, the relevance score $g_{l,o}^{(s)}$ was computed as:

$$g_{l,o}^{(s)} = \underset{c}{avg}\left(h_{l,o}^{(s)}\right) \Big/ \max_{l}\left(\underset{c}{avg}\left(h_{l,o}^{(s)}\right)\right), 0 \leq c \leq C - 1, \tag{6.8}$$

with $g_{l,o}^{(s)}$ being a scalar quantity $\in [0,1]$ summarizing the importance of the l-th band-pass filter for the o-th variable.

Subsequently, the frequencies belonging to the passband of the filter associated to the l-th feature map and defined by ${\theta_{spect,l}}^{(s)}$ (see (5)) were assigned to the corresponding relevance score $g_{l,o}^{(s)}$:

$$\begin{cases} p_l^{(s)}(f) = \begin{cases} 1, \ if\ f_{0,l}^{(s)} \leq f \leq f_{1,l}^{(s)} \\ 0, \ elsewhere \end{cases} \\ q_{l,o}^{(s)}(f) = g_{l,o}^{(s)} \cdot p_l^{(s)}(f) \end{cases}, \tag{6.9}$$

where $p_l^{(s)}(f)$ indicates the probability of a frequency f to be included in the passband of the l-th band-pass filter. Finally, the spectral relevance $q_o^{(s)}(f)$, quantifying the relevance of each frequency bin for the o-th kinematic variable, was obtained as:

$$q_o^{(s)}(f) = \underset{l}{\text{avg}}\, q_{l,o}^{(s)}(f). \tag{6.10}$$

From a preliminary analysis, the spectral relevance $q_o^{(s)}(f)$ resulted to be comparable across x- and y-axes for both position and velocity (permutation cluster test with threshold-free cluster enhancement [54], see Section 6.6.3 of Supplementary Materials). Therefore, the $q_o^{(s)}(f)$ was averaged along the axes, thus, obtaining only one average spectral relevance profile $r_o^{(s)}(f)$ for the position and one for the velocity, where the index o hereafter denotes the kinematic variable, i.e., $o \in \{p, v\}$:

$$\begin{cases} r_p^{(s)}(f) = \frac{q_{px}^{(s)}(f) + q_{py}^{(s)}(f)}{2} \\ r_v^{(s)}(f) = \frac{q_{vx}^{(s)}(f) + q_{vy}^{(s)}(f)}{2} \end{cases}. \tag{6.11}$$

Finally, the spectral relevance for each kinematic variable was averaged within EEG bands (hereafter named "EEG band relevance"), in the delta (0.18-4 Hz), theta (4-8 Hz), alpha (8-13 Hz), beta (13-30 Hz), and low-gamma (30-40 Hz) bands, so to identify the most relevant spectral features predicting the position or velocity.

Spectral clustering and spatial relevance computation

In a second processing stage, we performed clustering to reveal whether certain groups of subjects were sharing common neural signatures in the frequency domain, i.e., sharing similar patterns of relevance in the EEG rhythms. To do so, the EEG band relevance of both position and velocity was clustered using Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) [55], and the correlation between observations as distance metric. As HDBSCAN does not require to specify the number of clusters as input parameter, the optimal number of clusters according to correlation is automatically learned from the observations. The EEG band relevance was chosen as it summarizes the features of the spectral relevance profile $r_o^{(s)}(f)$ in a compact way (i.e., 2*5 features per subject, instead of 2*frequency bins per subject), being the clustering applied to a limited number of subjects (13 in this study).

For each cluster, the following processing was performed. At first, the spectral relevance was averaged across subjects defining the cluster. Then, the spatial relevance was computed as follows. Let us denote with $[f_{0,r}, f_{1,r}]$, $0 \le r \le N_{bands} - 1$ the r-th frequency range defining each EEG band (see previous step) and with $N_{bands} = 5$ the number of bands. For the s-th subject and for the r-th frequency range we considered the subset of the ICNN band-pass filters, denoted as $S_r^{(s)}$, containing in their passband the frequency bins belonging to $[f_{0,r}, f_{1,r}]$, and we extracted the spatial filters associated to this subset of band-pass filters, i.e., ${\theta_{spat,l}}^{(s)} = \left\{\theta_{lk}^{(s)}\right\}$ (see Equation 6.6), $l \in S_r^{(s)}, 0 \le k \le D_1^{ISS} - 1$. Spatial filters were considered in their absolute value, as done in [32,56]. Subsequently, the absolute spatial features were averaged

together, electrode per electrode ($\forall c, 0 \leq c \leq C-1$), and normalized to the maximum across electrodes, obtaining the spatial relevance:

$$\sigma_r^{(s)} = \underset{l \in S^{(s)},k}{\text{avg}} \, abs(\theta_{lk}^{(s)}) \Big/ \max_c \left(\underset{l \in S^{(s)},k}{\text{avg}} \, abs(\theta_{lk}^{(s)}) \right). \quad (6.12)$$

Lastly, $\sigma_r^{(s)}$ was averaged across subjects defining the cluster.

6.2.6. Performance metrics and state-of-the-art decoders

Once trained, the ICNN was evaluated on the EEG chunks belonging to the test set, obtaining the predicted trajectories of the 2-D position and velocity during each trial. The predicted trajectories were compared with the recorded ones by computing, for each subject, the Pearson's correlation coefficient (r) and the Root Mean Squared Error (RMSE). The proposed ICNN was compared with DeepConvNet and ShallowConvNet [29], representing two SOA CNNs performing single scale temporal feature learning but with a different depth: the first included 5 convolutional layers, while the second included 2 convolutional layers. These networks were originally proposed for motor imagery and execution classification from EEG signals sampled at 250 Hz. Therefore, we modified the activation function of the CNN last layer to solve regression instead of classification (i.e., linear instead of softmax activation). In addition, kernels (both convolutional and pooling) operating in the temporal domain were scaled down in their size by a factor of 2, due to the different sampling rate adopted in the original implementation, as done in [30,32]. Except for these changes, SOA CNNs were used with their original hyper-parameters, as proposed in Schirrmeister et al. [29]. Lastly, we compared our ICNN to a more traditional SOA algorithm, the non-linear PLS+UKF decoder as proposed in Kobler et al. [13], which was carefully designed to alleviate the amplitude mismatch problem characterizing linear decoders.

The same data preparation used for the ICNN (see Sections 6.2.1 and 6.2.2) was used for all SOA decoders, i.e., DeepConvNet, ShallowConvNet [29], and PLS+UKF [13]. In addition, the SOA CNNs were trained using the same training hyper-parameters (i.e., optimizer, learning rate, batch size, etc.) as those used for our ICNN, to provide a fair comparison.

6.2.7. Statistical analyses

The following statistical analyses were conducted.

i. To compare the proposed ICNN with the SOA, for each predicted trajectory (i.e., p_x, p_y, v_x, v_y), a pairwise comparison was performed between the performance obtained with MS-Sinc-ShallowNet and each SOA decoder, both trained using WS strategy (12 total tests).
ii. The performance obtained with MS-Sinc-ShallowNet trained with the WS strategy and with the LOSO strategy were compared via a pairwise comparison for each predicted trajectory (4 total tests).
iii. To compare the potential benefit in transferring the knowledge on a new subject from a pre-trained network on other subjects, for each trajectory and each number of training trials (see Section 6.2.4), a pairwise comparison was performed between MS-Sinc-ShallowNet trained using WS strategy and using TL-WS strategy (20 total tests).

iv. A Friedmann test was performed to compare the EEG band relevance across bands, separately for position and velocity. Then, as significant differences were found (see Section 6.3.2), post-hoc pairwise comparisons were performed testing all combinations (10 total tests), separately for position and velocity. This analysis was performed to evaluate which EEG band was the most relevant for decoding position and for decoding velocity.

Pairwise comparisons were performed using Wilcoxon signed-rank tests and false discovery rate correction at $\alpha = 0.05$ (Benjamini–Hochberg [57]) to correct for multiple tests.

6.3. RESULTS

6.3.1. Performance

Figure 6.3 reports the performance metrics obtained with MS-Sinc-ShallowNet and with SOA decoders, trained using the WS strategy, together with the results of the statistical analysis. When comparing MS-Sinc-ShallowNet to the PLS+UKF algorithm, the two decoders scored comparable Pearson's correlation coefficients for all the decoded trajectories. However, the proposed ICNN scored significantly lower RMSEs than the PLS+UKF algorithm for all decoded trajectories ($p < 0.05$), reflecting a better amplitude reconstruction, especially for the prediction of the velocity components ($p < 0.001$). Compared to the others SOA CNNs, MS-Sinc-ShallowNet significantly outperformed ShallowConvNet in both correlations and RMSEs ($p < 0.01$), and scored comparable correlations as DeepConvNet, though outperforming it as to the RMSEs in the x-axis.

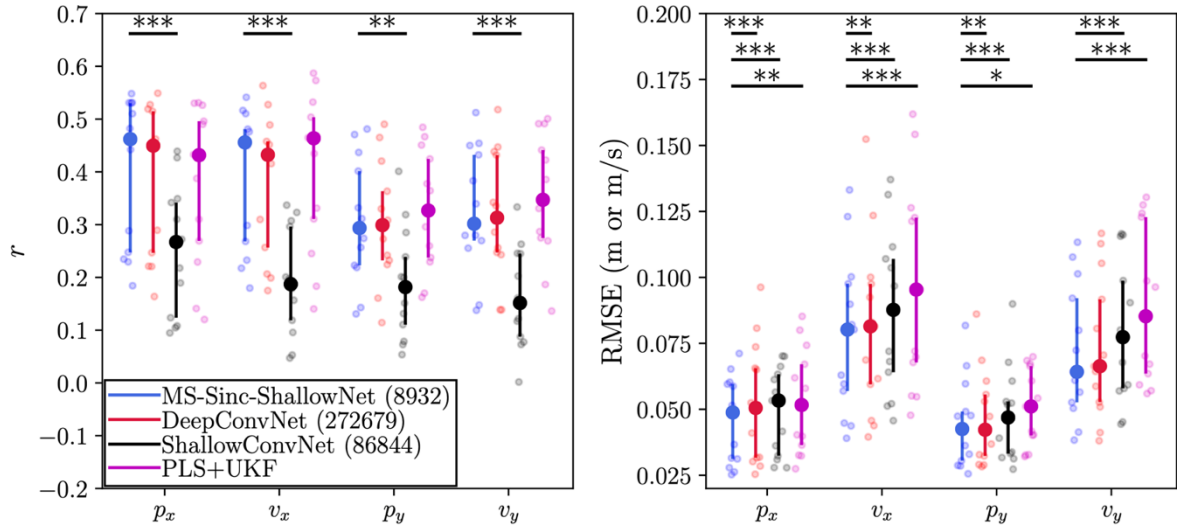


Figure 6.3 – Performance comparison between MS-Sinc-ShallowNet and state-of-the-art (SOA) decoders. Pearson's correlation coefficients (r, on the left) and RMSEs (on the right) for each decoded variable (p_x, p_y, v_x, v_y) scored with MS-Sinc-ShallowNet and SOA decoders, including DeepConvNet (red), ShallowConvNet (black), and PLS+UKF (magenta) are reported. All decoders were trained according to the within-subject strategy. The number of trainable parameters introduced in the CNNs is included within brackets in the legend. Smaller dots represent the performance metric scored for each subject, while bigger dots represent the median of each distribution and whiskers represent the 25th and 75th percentile. Significant corrected (Benjamini–Hochberg [57]) p-values are reported (*p<0.05, **p<0.01, ***p<0.001).

The decoding performance of MS-Sinc-ShallowNet was further investigated by adopting additional training strategies, such as the LOSO strategy and the TL-WS strategy. In Figure 6.4 the performance obtained with MS-Sinc-ShallowNet trained in LOSO is reported. As expected, due to the inter-subject EEG variability, significantly lower correlations were scored in LOSO compared to WS strategy ($p < 0.001$). However, LOSO models can be used to enable other training strategies such as TL-WS, where the knowledge learned from the other subjects is used as initial point for training the network on a new subject. Figure 6.5 shows the decoder performance, as measured by the Pearson's correlation coefficient, when TL-WS strategy was adopted, using a progressively increasing number of training trials that were randomly sampled from the training set of the held-out subject. Here, 2, 4, 6, 8, 10 training trials were randomly

sampled 10 times from the overall training set. For each reduced training set, the performance obtained by the WS decoders are reported together with the TL-WS performance, to highlight the potential benefit of transfer learning compared to the random initialization used in the WS strategy.

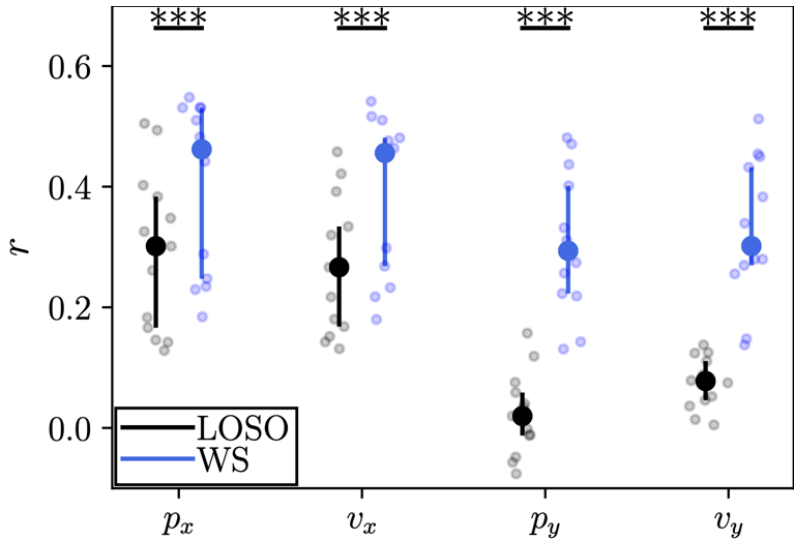


Figure 6.4 – Performance obtained with the leave-one-subject-out (LOSO) strategy. Pearson's correlation coefficient (r) is reported for each decoded variable obtained with MS-Sinc-ShallowNet trained in the LOSO strategy (black). The within-subject (WS) performance (blue) is reported too (same as the ones reported in Figure 6.3). Significant corrected (Benjamini–Hochberg [57]) p-values are reported (*p<0.05, **p<0.01, ***p<0.001).

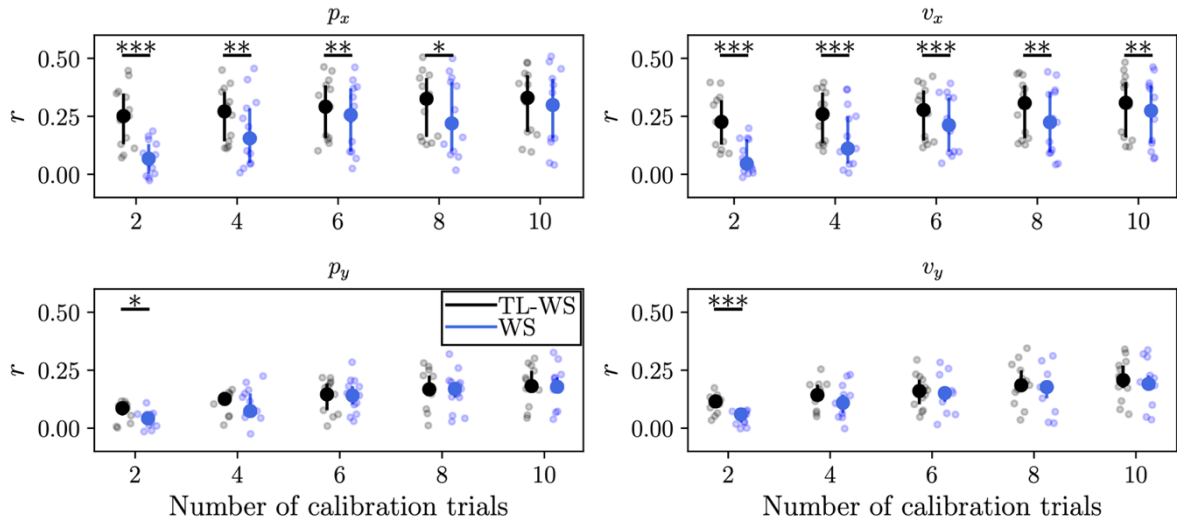


Figure 6.5 – Performance obtained while transferring the knowledge on single subjects (TL-WS). Pearson's correlation coefficient (r) is reported for each decoded variable (p_x, p_y, v_x, v_y) obtained when training MS-Sinc-ShallowNet with a more compact training set sampled from the original training set, by randomly sampling 2, 4, 6, 8, 10 trials by 10 times (see Section 6.2.4). MS-Sinc-ShallowNet was trained by adopting a random initialization (black), corresponding to the WS strategy, or importing weights from a pre-trained network with LOSO strategy (blue), corresponding to the TL-WS strategy. Significant corrected (Benjamini–Hochberg [57]) p-values are reported (*p<0.05, **p<0.01, ***p<0.001).

Remarkably, transfer learning was found to be widely beneficial across the decoded variables (p_x, p_y, v_x, v_y), especially when using extremely compact training sets, e.g., with only 2, 4 training trials, providing significant improvements in performance for all variables.

Furthermore, significant improvements were maintained also when using more training trials (e.g., 6, 8, 10) for the position and velocity along the x-axis.

6.3.2. Spectral and spatial relevance related to position and velocity

The spectral relevance and the EEG band relevance are reported in Figure 6.6 respectively within the top and bottom panels. The frequency components in the delta band are the ones with highest relevance, for both position and velocity. This is further confirmed by the statistical analysis of the EEG band relevance: for both position and velocity, significant differences were found in the EEG band relevance across bands ($p < 0.001$, Friedmann test) and, in particular, these significant differences resulted between delta and each of the other bands (theta, alpha, beta, low-gamma, $p < 0.001$ post-hoc pairwise tests).

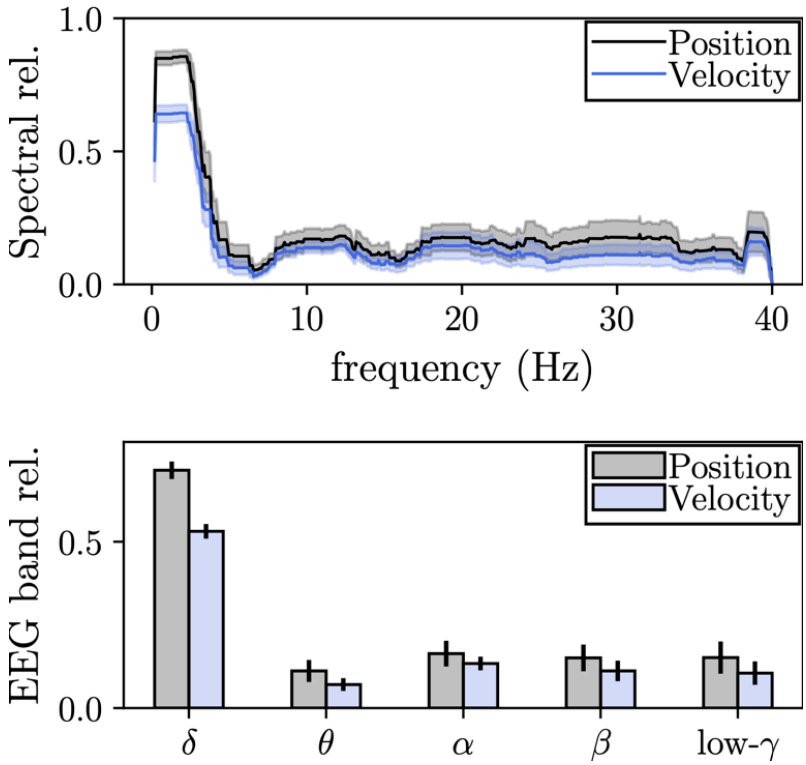


Figure 6.6 – Spectral relevance (top) and EEG band relevance (bottom) of position (black) and velocity (blue), as obtained with the proposed ICNN+ET analysis. Mean and standard error of the mean across subjects of the spectral relevance (thick line and shaded area) and of the EEG band relevance (height of the bars and error bar) are reported.

When clustering the subjects based on EEG band relevance two clusters were obtained from the HDBSCAN algorithm (cluster 0 with 6 subjects, cluster 1 with 7 subjects). The clusters gave information on the most common strategies picked by the ICNN in the frequency domain to decode position and velocity. From Figure 6.7, it is evident that the delta band had the highest relevance in both clusters, meaning that the delta band was widely exploited across subjects. Nevertheless, subjects in cluster 0 additionally exhibited relevance of higher frequency ranges such as alpha, beta and low-gamma, for the decoding task. Regarding the spatial relevance, this was investigated separately for each cluster. According to the results of spectral relevance analysis (upper panels of Figure 6.7), we considered the spatial relevance in the delta range for

both clusters and, in addition, in the alpha, beta and low-gamma ranges for cluster 0. In these EEG bands, the most relevant electrodes to decode hand position and velocity covered the contralateral central/centro-parietal and parietal/parieto-occipital sites.

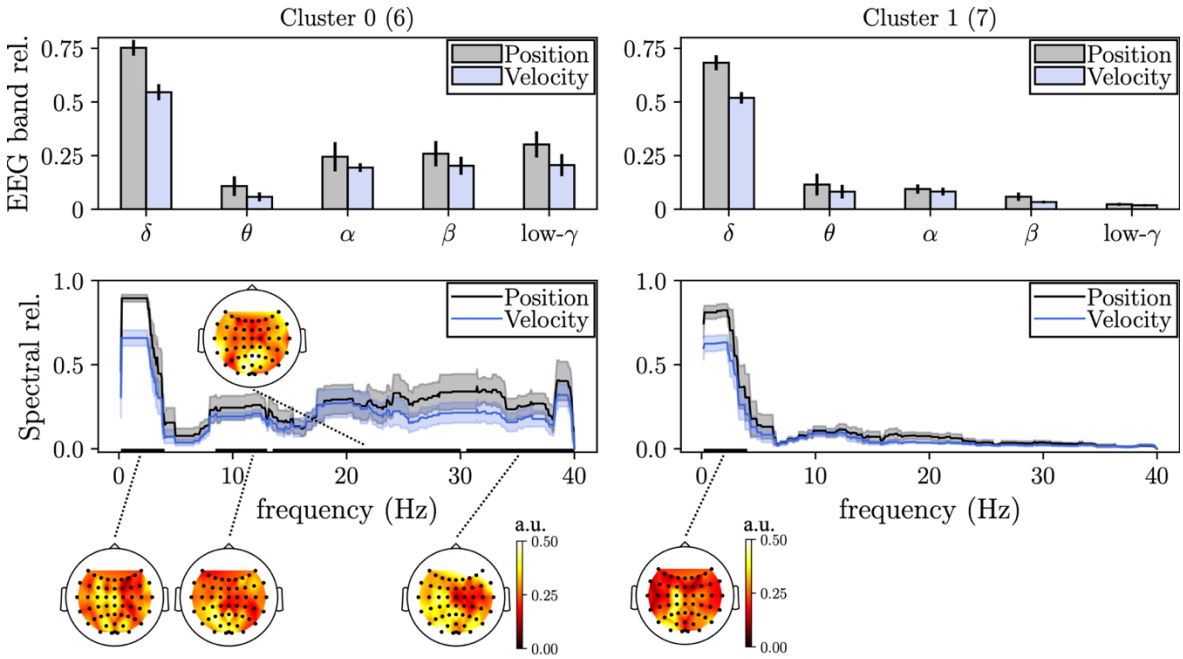


Figure 6.7 – Cluster analysis of spectral features and spatial features encoding kinematics. Two clusters were obtained: cluster 0 (left) with 6 subjects, and cluster 1 (right) with 7 subjects. The number of subjects is reported within brackets. For each cluster, mean and standard error of the mean across subjects of the EEG band relevance (height of the bars and error bar, on top) and of the spectral relevance (thick line and shaded area, on bottom) are reported. Lastly, the spatial relevance linked to the analyzed EEG bands (black lines on the x-axis) is displayed.

6.4. DISCUSSION

In this study, a light and interpretable CNN was designed and validated for EEG-based trajectory decoding of 2-D position and velocity during a pursuit tracking task. The ICNN was designed to learn interpretable spectral and spatial features in its first layers, while learning deep temporal features at two different time scales. ICNN layers were designed to ensure a limited model size, e.g., by adopting interpretable, depthwise and separable convolutions. Conversely to the SOA, in which trajectory decoding is mostly performed by exploiting LF-EEG, in this study the learning system was left free to explore also higher frequencies EEG components, potentially encoding kinematic information. The proposed light ICNN was compared to other SOA CNNs [29], previously validated for motor classification and adapted here for trajectory decoding, and to the SOA non-linear PLS+UKF algorithm proposed for trajectory decoding by Kobler et al. [13]. Furthermore, the ICNN was trained with different strategies including within-subject, leave-one-subject-out, and transfer learning, to test the potentialities of the ICNN under different practical scenarios involving more variable or compact calibration sets. Lastly, the increased interpretability of ICNN spectral and spatial features, was coupled with an ET to highlight the most relevant neural signatures encoding position and velocity, as learned by the learning system in a data-driven way.

6.4.1. Performance

The correlations between the predicted and actual hand kinematics obtained via the proposed ICNN were comparable to those obtained via the traditional non-linear PLS+UKF decoder, but with a significantly better amplitude reconstruction, as denoted by the significantly lower RMSEs scored by the ICNN (see Figure 6.3). This is of relevance as, despite the adopted non-linear PLS+UKF was designed to alleviate the amplitude mismatch problem in trajectory decoding [13], the proposed non-linear decoder based on an interpretable deep neural network provided a better estimation of both positions and velocities.

Compared to the other tested SOA CNNs, our CNN was lighter, introducing only about 9K trainable parameters compared to about 273K and 87K trainable parameters of DeepConvNet and ShallowConvNet, respectively, and resulting in a design more suitable for small datasets. In addition, our CNN by integrating layers devoted to ease the interpretability, i.e., employing an interpretable spectral and spatial feature extractor based on sinc- and depthwise convolutions, resulted more interpretable in the learned spectral and spatial features than SOA CNNs that employ standard convolutions. The increased interpretability was implemented by exploiting a re-parametrization that limits the ICNN to explore only band-pass filters in the temporal domain. Despite this solution reduced the capacity of the learning system (i.e., the ability of approximating a wide variety of functions), the ICNN outperformed ShallowConvNet both in terms of correlations and RMSEs, and performed similarly or slightly better than DeepConvNet. Therefore, overall, the proposed decoder performed on par or even outperformed heavier and less interpretable CNN designs, representing a better trade-off between performance, interpretability, and model size.

Lastly, it is worth noticing that for all decoders correlations were lower for trajectories predicted in the y-axis than in the x-axis, as found in a previous study on a subset of the adopted dataset [22]. This might be related to the experimental setup, where the screen was tilted

towards the y-axis to facilitate the movements of the robot. Thus, the perception of movements of the moving object along the y-axis may be ambiguous in comparison with the ones along the x-axis [22].

From the sensitivity analysis on the hyper-parameters (see Section 6.6.2 of Supplementary Materials), a reduction in the number of band-pass filters and spatial filters resulted in general detrimental in terms of performance. Conversely, by increasing the number of filters, only small performance improvements were observed, suggesting that the adopted ICNN architecture (see Section 6.2.3 and Supplementary Table 6.1 for details about the hyper-parameters) already had enough capacity for trajectory decoding. Interestingly, despite CNN-based motor classification proved that batch normalization provides a significant improvement in performance [29,30,32,47], this technique worsened the performance in the trajectory decoding problem addressed in this study. This may be due to low capacity of the ICNN, which might require less regularization, and to the different nature of the decoding problem. Indeed, batch normalization was mainly adopted in the literature in classification problems from single EEG trials [29,30,32,47], while in this study we aimed to solve a regression problem from overlapped EEG chunks extracted from EEG trials.

The ICNN was further evaluated by testing its ability to transfer the knowledge learned from other subjects to a held-out subject, with the aim of reducing the training time of the decoder. To perform transfer learning, an architecture pre-trained on other subjects (different from the held-out one) was used, corresponding to the LOSO model (see Figure 6.4). TL-WS results (Figure 6.5) highlighted a significant increase in decoding performance when using transfer learning (up to a median increase of 0.18 in correlations), especially in case of the lowest data regime (e.g., 2, 4 training trials). Therefore, the LOSO model, by capturing relevant cross-subject features, resulted a significantly better initialization point in the parameter space than the random one, for training the network on a new subject. These results suggest that the proposed interpretable non-linear decoder, besides improving amplitude reconstruction compared to SOA non-linear PLS+UKF, was also capable of transferring the knowledge from other subjects, enabling its usage with extremely compact-sized training sets. This could have prospective implication for a practical usage of the decoder in BCI systems, thanks to the potentiality of TL of reducing training times during BCI sessions.

6.4.2. Spectral and spatial relevance related to position and velocity

We took advantage of the proposed ICNN combined with ET (ICNN+ET algorithm) to highlight the relevant neural signatures encoding position and velocity in the frequency domain. For both kinematic variables, the delta band resulted to be the most relevant EEG band (see Figure 6.6), while higher frequency ranges (e.g., alpha, beta and low-gamma) appeared to be relevant, in addition to delta, with more variability across subjects. The involvement of these bands in addition to the delta band might reflect the involvement of sensorimotor rhythms. Specifically, the contributions of higher frequency ranges emerged while analyzing the spectral relevance and the EEG band relevance at the level of each cluster of subjects. As reported in Figure 6.7, the relevance for the delta band resulted the highest for both clusters, and the two clusters of subjects did not differ for the delta contribution; however, one of the two clusters (cluster 0) additionally showed higher relevance values for higher frequencies, like alpha, beta

and low-gamma ranges, with respect to the second (cluster 1). This result suggests that the ICNN widely exploited the delta band across all subjects, while the contribution of higher frequency ranges to solve the decoding problem was modulated depending on the subject-specific neural signatures.

The highest relevance found for the delta band agrees with what has been reported in literature, supporting the hypothesis that the low-frequency (< 3 Hz) band of the EEG overall contains highly relevant information for the decoding of voluntary movement [14–16,18,19,21,22], and widely across subjects. The results further suggest that higher frequency ranges like beta and low-gamma also have a role and carry information about the movement, although the extent of their contribution is more variable across subjects. This is in line to what already was observed in [20], where circular arm movements were decoded from both low-frequency amplitude features and higher-frequency power features. In particular, while the trajectories could be reconstructed from all subjects when using the low-frequency features, they could not always be successfully estimated when using the higher-frequency components alone [20].

Previous decoding studies suggested how the kinematic information can be best decoded from amplitude features in the low-frequency range, however, power features should be used for higher frequencies [16,17,20], as they likely reflect the well-known modulation of sensorimotor rhythms with voluntary movement. Provided that, in our approach, the network takes as input the amplitude of the signal in a wider frequency range, it might appear like the amplitude features only are used, independently of the frequency content. It should, however, be noted that the CNN generally approximates a non-linear function, which may produce an equivalent effect of computing the power of the signal. Therefore, it might not be excluded that non-linear features extracted from the signal (equivalent, for example, to computing the power) are being exploited by the network at higher frequency components, with the advantage that the band-width of the filters is not to be determined “a-priori”, but is automatically learned by the network, according to the most relevant and subject-specific content to solve the decoding task.

Through the spatial relevance analysis, we examined which electrodes were more important for encoding position and velocity inside relevant EEG bands. The electrode sites encoding more kinematics were the ones approximately covering the sensorimotor and parieto-occipital area. Especially, across frequency bands, the most relevant electrodes were covering the contra-lateral (i.e., left) primary sensorimotor areas (Figure 6.7), therefore likely reflecting the modulations of sensorimotor rhythms accompanying voluntary movement, and in line with the findings of previous studies [17,20,58]. Nevertheless, along with the sensorimotor areas, the ICNN was found to rely also on parieto-occipital sites to decode the kinematics, similarly to the findings of [15,21,22] for the delta band, and [16,17,20] for the beta band. This could be explained by the nature of the task, which not only involved the hand movement, but also visual processing and eye-hand movement coordination [59].

6.5. CONCLUSIONS

In this study we proposed a novel light and dual-scale ICNN for trajectory decoding, that learns interpretable spectral and spatial features, and deep temporal features at two time scales. At variance with the SOA of trajectory decoding, the ICNN was not only restricted to using low frequency EEG but was left free to exploit also higher frequencies. The ICNN provided a significant better amplitude reconstruction compared to a more traditional non-linear decoder based on PLS+UKF. Thus, future non-linear decoders for trajectory decoding may benefit from using the proposed ICNN to better reconstruct the amplitude of movements, and to provide a more natural control of actuators in BCIs. Furthermore, the proposed ICNN performed on par or even outperformed other SOA CNNs, at the same time providing a more interpretable spectral and spatial feature learning and a lighter design. Therefore, compared to SOA CNNs, our decoder resulted a better compromise between interpretability, performance and model size.

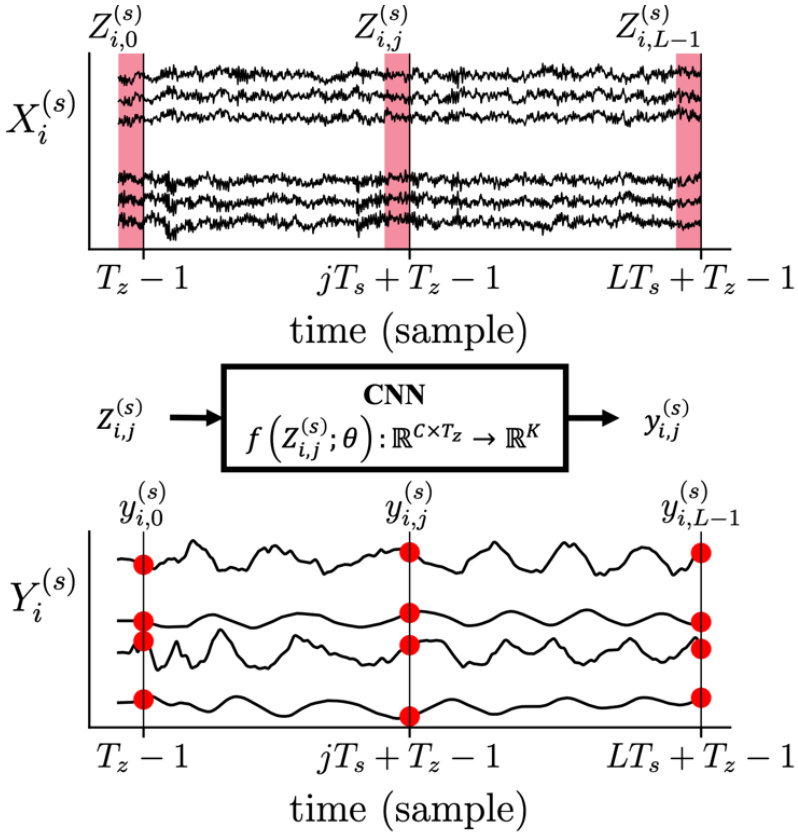
In addition, transfer learning significantly improved the performance especially when using few training trials and, thus, could enable a reduction of BCI calibration times in future studies. All these results contribute to the development of accurate and plug-and-play decoders for trajectory decoding.

Next, in this study we exploited the interpretable structure of the proposed non-linear decoder to design an ICNN+ET algorithm devoted to analyzing the spectral and spatial features that mostly encode position and velocity. This analysis enabled to leverage the automatic feature learning of the ICNN to provide useful data-driven representations in the frequency and spatial domain about the underlying neural processes. The analysis of spectral features widely matched across subjects the well-known delta relevance for trajectory decoding. However, in addition to delta, the results of the ICNN+ET analysis confirmed that for some subjects also alpha, beta and low-gamma may encode relevant position and velocity information for decoding, although this second feature is exploited less consistently across subjects. Lastly, the analysis of spatial features showed that sensorimotor and parieto-occipital sites most encode the position and velocity, in line with previous studies and with the visuomotor nature of the task. All these results suggest that the ICNN+ET algorithm was capable of characterizing the neural encoding of position and velocity, and that future studies may benefit from using the proposed ICNN+ET algorithm to analyze the most relevant neural features related to kinematics.

6.6. SUPPLEMENTARY MATERIALS

6.6.1. Continuous trajectory decoding from the EEG

To perform continuous decoding of 2-D positions and velocities, the CNN input was a buffer of EEG data ("EEG chunk"), consisting of T_z time samples of the multi-variate input time series recorded during each trial (see Section 6.2.2), extracted with a stride of T_s samples. This procedure is reported in the Supplementary Figure 6.1 and described in Section 6.2.1.



Supplementary Figure 6.1 – Schematic of the extraction of EEG chunks and trajectory values from EEG trials. Each red shaded area (top) denotes the buffer of EEG data forming the EEG chunk, and its 4 associated red dots (bottom) denote the kinematic values.

6.6.2. Hyper-parameter search and hyper-parameter sensitivity analysis

To select the optimal hyper-parameters defining the architecture and to train the architecture, Bayesian optimization [51] was performed while training within-subject models. The searched hyper-parameters were ${F_0}^{ISS}$, ${K_0}^{ISS}$, ${D_1}^{ISS}$, the number of parallel branches each learning features at a different time scale (N_s), ${K_1}^{DST}$, use of batch normalization [50], p, and the learning rate. The meaning of these symbols is described in the Sections 6.2.1 and 6.2.3. Batch normalization [50] normalizes the network intermediate outputs, speeding up training, reducing the influence of a specific parameter initialization scheme, and introducing a regularizer effect. SOA CNNs widely exploit this normalization for motor classification problems (e.g., [29,30,32,47]). To evaluate whether this technique could be useful also for trajectory decoding, batch normalization was applied to the output of each convolutional layer before the activation function, as suggested in [50]. The following procedure was performed

to select the kernel sizes of the multi-scale temporal feature extractor. At first, the kernel size of the largest scale was set the same as the searched value of ${F_0}^{ISS}$, denoting the temporal kernel size of the first layer. Then, depending on the number of scales selected, the kernel size of the i-th parallel branch ($1 \leq i \leq N_s$) is defined automatically as $(1, {F_0}^{ISS}[1]/i)$, $1 \leq i \leq N_s$, taking the nearest odd number, e.g., when ${F_0}^{ISS} = (1,51)$, then kernel sizes of the multiple scales are set to $(1,51)$ and $(1,25)$ if $N_s = 2$, or $(1,51)$, $(1,25)$, $(1,17)$ if $N_s = 3$. Bayesian optimization was performed for 100 iterations using the validation loss as metric to minimize, tree-structured Parzen estimator as surrogate function and expected improvement as selection function. The optimal configuration of the ICNN was selected as the most frequent one across the subject-specific Bayesian-optimized models and corresponds to the one described in Supplementary Table 6.1. Details on the source space are reported in Supplementary Table 6.2.

Supplementary Table 6.1 –MS-Sinc-ShallowNet. Each layer is provided with its name, main hyper-parameters, number of trainable parameters, and output shape. See Sections 6.2.1 and 6.2.3 for the meaning of the symbols. In all layers, where not specified, stride (S) and padding (P) were set to $(1,1)$ and $(0,0)$, respectively.

Block	Layer name	Hyper-parameters	Number of tr. parameters	Output shape
	Input	${K_0} = 1$	0	(1,53,100)
ISS	Sinc-Conv2D	${K_0}^{ISS} = 16$, ${F_0}^{\mathrm{ISS}} = (1,51)$, ${P_0}^{\mathrm{ISS}} = (0,25)$	32	(16,53,100)
	Depthwise-Conv2D	${D_1}^{ISS} = 2$, ${K_1}^{ISS} = 32$, ${F_1}^{ISS} = (53,1)$, $c = 1$	1728	(32,1,100)
	ELU		0	(32,1,100)
	Dropout	$p = 0.5$	0	(32,1,100)
DST-large scale	Separable (Depth.+Point.)-Conv2D	${K_1}^{DST} = 32$, ${F_0}^{DST} = (1,51)$, ${D_0}^{DST} = 1$, ${P_0}^{DST} = (0,25)$	2720 (1664+1056)	(32,1,100)
	ELU		0	(32,1,100)
	AvgPool2D	${F_p}^{DST} = (1,10)$	0	(32,1,10)
	Dropout	$p = 0.5$	0	(32,1,10)
DST-short scale	Separable (Depth.+Point.)-Conv2D	${K_1}^{DST} = 32$, ${F_0}^{DST} = (1,25)$, ${D_0}^{DST} = 1$, ${P_0}^{DST} = (0,12)$	1888 (832+1056)	(32,1,100)
	ELU		0	(32,1,100)
	AvgPool2D	${F_p}^{DST} = {S_p}^{DST} = (1,10)$	0	(32,1,10)
	Dropout	$p = 0.5$	0	(32,1,10)
Regressor	Concatenate		0	(64,1,10)
	Flatten		0	(640)
	Fully-Connected	$N = K$, $c = 1$	2564	(4)
			8932	

Then, the main hyper-parameters of the proposed ICNN were investigated by changing one hyper-parameter at a time and evaluating the change in the performance, i.e., performing a sensitivity analysis on the hyper-parameters, as performed to study hyper-parameters in [29,32,33]. Hereafter, the ICNN with the baseline hyperparameters defined in Supplementary Table 6.1, will be referred as "baseline" architecture. Then, we changed the value of one hyper-parameter at a time of the baseline architecture, realizing a "variant" architecture. Both architectural (i.e., parameters affecting the overall architecture design) and training hyper-

parameters were investigated (i.e., parameters influencing the training). These were:

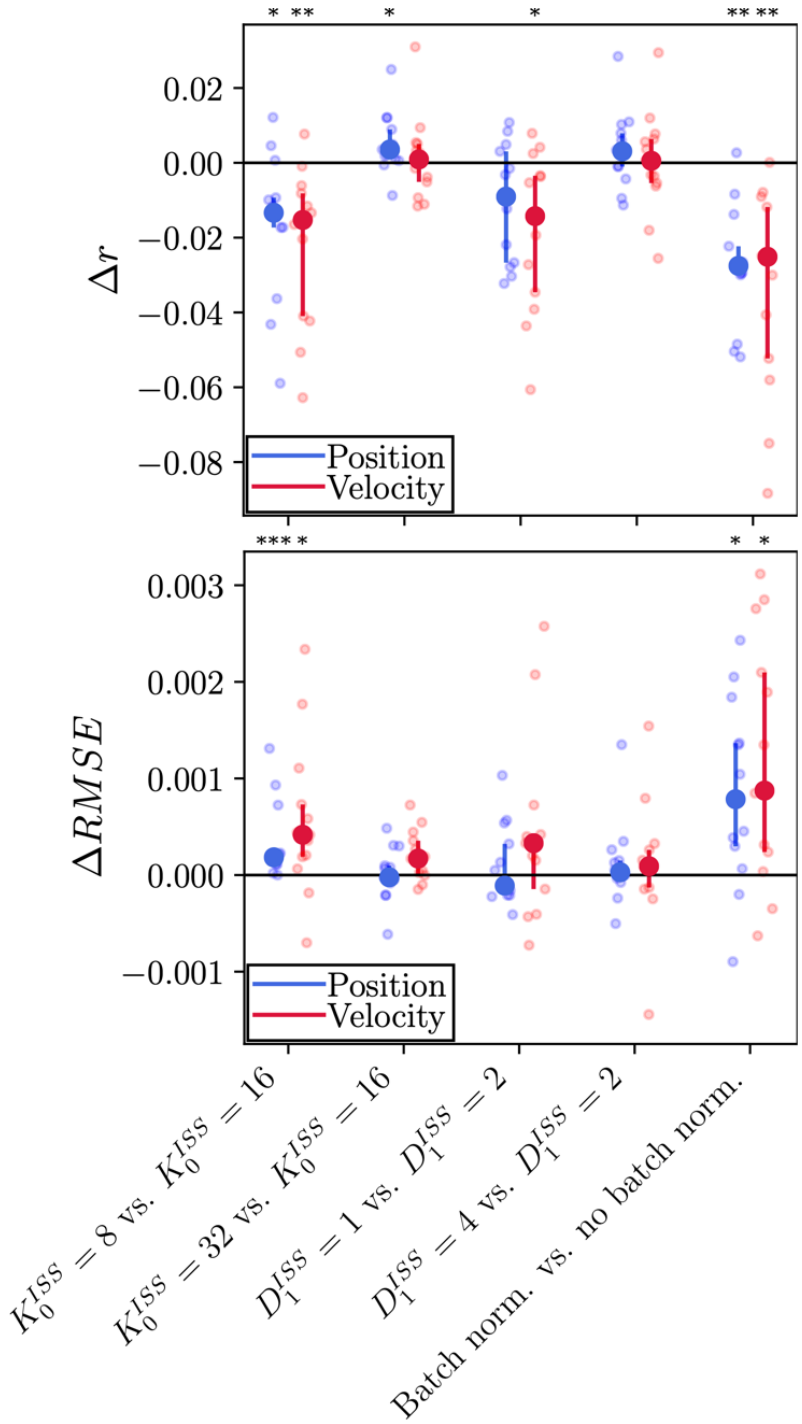
i. The number of trainable band-pass filters in Block 1 (${K_0}^{ISS}$). In the baseline architecture, 16 band-pass filters were learned. To investigate designs using less or more filters, two variant architectures were developed, by setting ${K_0}^{ISS} = 8$ or ${K_0}^{ISS} = 32$. Of course, using less or more filters is associated to a reduced or increased model size, respectively.
ii. The number of spatial filters tied to each band-pass filter in Block 1 (${D_1}^{ISS}$). In the baseline architecture, 2 spatial filters were learned for each band-pass filter. To investigate designs using less or more spatial filters, two variant architectures were developed, by setting ${D_1}^{ISS} = 1$ or ${D_1}^{ISS} = 4$. Of course, using less or more filters is associated to a reduced or increased model size, respectively.
iii. The inclusion of batch normalization [50]. In the baseline architecture, batch normalization was not adopted. To evaluate whether this technique could be useful also for trajectory decoding, batch normalization was included in the architecture, as specified at the beginning of Section 6.6.2 of Supplementary Materials.

Supplementary Table 6.2 – Searched hyper-parameters of MS-Sinc-ShallowNet: distributions and admitted values sampled during Bayesian optimization. Curly brackets denote discrete admitted values, while square brackets denote interval of admitted values.

Hyper-parameter	Distribution	Values
${F_0}^{ISS}$	uniform	{(1,25), (1,51)}
${K_0}^{ISS}$	uniform	{8,16,32,64}
${D_1}^{ISS}$	uniform	{1,2,4}
N_s	uniform	{1,2,3}
${K_1}^{DST}$	uniform	$\{{K_0}^{ISS} \cdot {D_1}^{ISS}, 1, 2, 4\}$
Use of batch norm.	uniform	{False, True}
p	uniform	{0, 0.25, 0.5}
Learning rate	log-uniform	[1e-5, 1e-2]

Performance metrics were computed for these variant conditions as described in Section 6.2.6. The performance difference between each of the previously described variants and the baseline architecture was computed. Then, for brevity the performance difference was averaged across x- and y-axis components, and the effect of each hyper-parameter was studied for position and velocity, separately. The performance of the baseline MS-Sinc-ShallowNet was compared with the performance obtained with each variant MS-Sinc-ShallowNet design with pairwise comparisons (10 total tests). Pairwise comparisons were performed using Wilcoxon signed-rank tests and false discovery rate correction at $\alpha = 0.05$ (Benjamini–Hochberg [57]) to correct for multiple tests.

Correlation and RMSE differences between variant ICNNs and the baseline ICNN are reported in Supplementary Figure 6.2, together with the results of the statistical analysis.



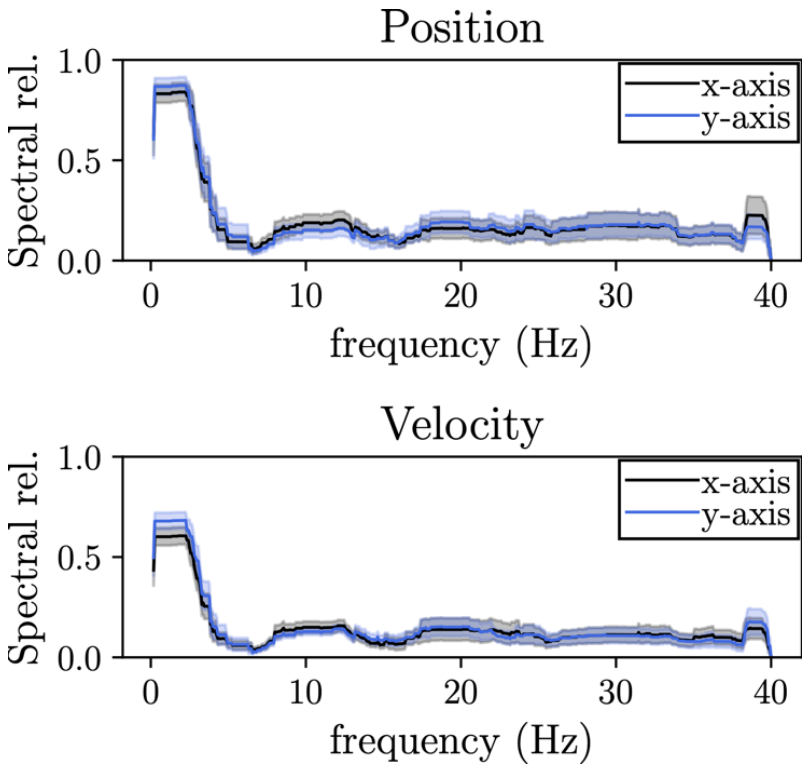
Supplementary Figure 6.2 – Sensitivity analysis on the main ICNN hyper-parameters. Difference of the Pearson's correlation coefficients (Δr in Supplementary Figure 6.2A) and of the RMSE ($\Delta RMSE$, in Supplementary Figure 6.2B) between each MS-Sinc-ShallowNet variant and the baseline MS-Sinc-ShallowNet, both trained in within-subject (WS) strategy. Smaller dots represent the performance difference for each subject, while bigger dots represent the median of each distribution and whiskers represent the 25th and 75th percentile. Significant corrected (Benjamini–Hochberg [57]) p-values are reported (*p<0.05, **p<0.01, ***p<0.001).

In addition to significant higher RMSE ($p < 0.05$), significantly lower correlations were obtained when using a lower number of band-pass filters in Block 1 ($p < 0.05$), and when including batch normalization ($p < 0.01$), for both position and velocity. Lower correlations ($p < 0.05$) were also found when using less spatial filters in Block 1, but with a significant

worsening in performance only for velocity component. Using more filters in Block 1 (both band-pass and spatial filters) only slightly increased the performance, with a small but significant ($p < 0.05$) improvement for position correlations.

6.6.3. Spectral relevance comparison across directions

In this study, the spectral relevance was computed for each decoded trajectory, i.e., p_x, p_y, v_x, v_y, see Equation 6.10, and is reported in Supplementary Figure 6.3 for each coordinate. A permutation cluster test with 5000 permutations and by using threshold-free cluster enhancement [54] was performed between the relevance values in the x- and y-axes, separately for position and velocity. This was done to test whether there are significant differences between the two different directions. No significant differences were obtained, highlighting that the spectral relevance between the two directions was comparable.



Supplementary Figure 6.3 – Spectral relevance for position and velocity. The spectral relevance is reported in its mean value (tick line) and standard error of the mean (shaded area) across subjects.

6.7. REFERENCES

[1] Wolpaw J R, Birbaumer N, McFarland D J, Pfurtscheller G and Vaughan T M 2002 Brain–computer interfaces for communication and control *Clinical neurophysiology* **113** 767–91

[2] Millán J D R 2010 Combining brain-computer interfaces and assistive technologies: state-of-the-art and challenges *Front. Neurosci.* **1**

[3] Müller-Putz G R, Kobler R J, Pereira J, Lopes-Dias C, Hehenberger L, Mondini V, Martínez-Cagigal V, Srisrisawang N, Pulferer H, Batistić L and Sburlea A I 2022 Feel Your Reach: An EEG-Based Framework to Continuously Detect Goal-Directed Movements and Error Processing to Gate Kinesthetic Feedback Informed Artificial Arm Control *Front. Hum. Neurosci.* **16** 841312

[4] Bouton C E, Shaikhouni A, Annetta N V, Bockbrader M A, Friedenberg D A, Nielson D M, Sharma G, Sederberg P B, Glenn B C, Mysiw W J, Morgan A G, Deogaonkar M and Rezai A R 2016 Restoring cortical control of functional movement in a human with quadriplegia *Nature* **533** 247–50

[5] Collinger J L, Wodlinger B, Downey J E, Wang W, Tyler-Kabara E C, Weber D J, McMorland A J, Velliste M, Boninger M L and Schwartz A B 2013 High-performance neuroprosthetic control by an individual with tetraplegia *The Lancet* **381** 557–64

[6] Pistohl T, Ball T, Schulze-Bonhage A, Aertsen A and Mehring C 2008 Prediction of arm movement trajectories from ECoG-recordings in humans *Journal of Neuroscience Methods* **167** 105–14

[7] Schalk G, Kubánek J, Miller K J, Anderson N R, Leuthardt E C, Ojemann J G, Limbrick D, Moran D, Gerhardt L A and Wolpaw J R 2007 Decoding two-dimensional movement trajectories using electrocorticographic signals in humans *J. Neural Eng.* **4** 264–75

[8] Hochberg L R, Bacher D, Jarosiewicz B, Masse N Y, Simeral J D, Vogel J, Haddadin S, Liu J, Cash S S, van der Smagt P and Donoghue J P 2012 Reach and grasp by people with tetraplegia using a neurally controlled robotic arm *Nature* **485** 372–5

[9] Yeom H G, Kim J S and Chung C K 2013 Estimation of the velocity and trajectory of three-dimensional reaching movements from non-invasive magnetoencephalography signals *J. Neural Eng.* **10** 026006

[10] Waldert S, Preissl H, Demandt E, Braun C, Birbaumer N, Aertsen A and Mehring C 2008 Hand Movement Direction Decoded from MEG and EEG *Journal of Neuroscience* **28** 1000–8

[11] Georgopoulos A P, Langheim F J P, Leuthold A C and Merkle A N 2005 Magnetoencephalographic signals predict movement trajectory in space *Exp Brain Res* **167** 132–5

[12] Bradberry T J, Rong F and Contreras-Vidal J L 2009 Decoding center-out hand velocity from MEG signals during visuomotor adaptation *NeuroImage* **47** 1691–700

[13] Kobler R J, Sburlea A I, Mondini V, Hirata M and Müller-Putz G R 2020 Distance- and speed-informed kinematics decoding improves M/EEG based upper-limb movement decoder accuracy *J. Neural Eng.* **17** 056027

[14] Bradberry T J, Gentili R J and Contreras-Vidal J L 2010 Reconstructing Three-Dimensional Hand Movements from Noninvasive Electroencephalographic Signals *Journal of Neuroscience* **30** 3432–7

[15] Kobler R J, Sburlea A I and Müller-Putz G R 2018 Tuning characteristics of low-frequency EEG to positions and velocities in visuomotor and oculomotor tracking tasks *Sci Rep* **8** 17713

[16] Lv J, Li Y and Gu Z 2010 Decoding hand movement velocity from electroencephalogram signals during a drawing task *BioMed Eng OnLine* **9** 64

[17] Korik A, Sosnik R, Siddique N and Coyle D 2018 Decoding Imagined 3D Hand Movement Trajectories From EEG: Evidence to Support the Use of Mu, Beta, and Low Gamma Oscillations *Front. Neurosci.* **12** 130
[18] Úbeda A, Azorín J M, Chavarriaga R and R. Millán J del 2017 Classification of upper limb center-out reaching tasks by means of EEG-based continuous decoding techniques *J NeuroEngineering Rehabil* **14** 9
[19] Úbeda A, Hortal E, Iáñez E, Perez-Vidal C and Azorín J M 2015 Assessing Movement Factors in Upper Limb Kinematics Decoding from EEG Signals ed S J Bensmaia *PLoS ONE* **10** e0128456
[20] Kobler R J, Almeida I, Sburlea A I and Müller-Putz G R 2020 Using machine learning to reveal the population vector from EEG signals *J. Neural Eng.* **17** 026002
[21] Mondini V, Kobler R J, Sburlea A I and Müller-Putz G R 2020 Continuous low-frequency EEG decoding of arm movement for closed-loop, natural control of a robotic arm *J. Neural Eng.* **17** 046031
[22] Martinez-Cagigal V, Kobler R J, Mondini V, Hornero R and Muller-Putz G R 2020 Non-linear online low-frequency EEG decoding of arm movements during a pursuit tracking task *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)* 2020 42nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC) in conjunction with the 43rd Annual Conference of the Canadian Medical and Biological Engineering Society (Montreal, QC, Canada: IEEE) pp 2981–5
[23] Waldert S, Pistohl T, Braun C, Ball T, Aertsen A and Mehring C 2009 A review on directional information in neural signals for brain-machine interfaces *Journal of Physiology-Paris* **103** 244–54
[24] Robinson N and Vinod A P 2016 Noninvasive Brain-Computer Interface: Decoding Arm Movement Kinematics and Motor Control *IEEE Syst. Man Cybern. Mag.* **2** 4–16
[25] Ofner P and Muller-Putz G R 2015 Using a Noninvasive Decoding Method to Classify Rhythmic Movement Imaginations of the Arm in Two Planes *IEEE Trans. Biomed. Eng.* **62** 972–81
[26] Roy Y, Banville H, Albuquerque I, Gramfort A, Falk T H and Faubert J 2019 Deep learning-based electroencephalography analysis: a systematic review *Journal of Neural Engineering* **16** 051001
[27] Craik A, He Y and Contreras-Vidal J L 2019 Deep learning for electroencephalogram (EEG) classification tasks: a review *J. Neural Eng.* **16** 031001
[28] Lindsay G 2020 Convolutional Neural Networks as a Model of the Visual System: Past, Present, and Future *Journal of Cognitive Neuroscience* 1–15
[29] Schirrmeister R T, Springenberg J T, Fiederer L D J, Glasstetter M, Eggensperger K, Tangermann M, Hutter F, Burgard W and Ball T 2017 Deep learning with convolutional neural networks for EEG decoding and visualization *Human brain mapping* **38** 5391–420
[30] Lawhern V J, Solon A J, Waytowich N R, Gordon S M, Hung C P and Lance B J 2018 EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces *Journal of Neural Engineering* **15** 056013
[31] Simões M, Borra D, Santamaría-Vázquez E, GBT-UPM, Bittencourt-Villalpando M, Krzemiński D, Miladinović A, Neural_Engineering_Group, Schmid T, Zhao H, Amaral C, Direito B, Henriques J, Carvalho P and Castelo-Branco M 2020 BCIAUT-P300: A Multi-Session and Multi-Subject Benchmark Dataset on Autism for P300-Based Brain-Computer-Interfaces *Front. Neurosci.* **14** 568104
[32] Borra D, Fantozzi S and Magosso E 2020 Interpretable and lightweight convolutional neural network for EEG decoding: Application to movement execution and imagination *Neural Networks* S0893608020302021

[33] Borra D, Fantozzi S and Magosso E 2021 A Lightweight Multi-Scale Convolutional Neural Network for P300 Decoding: Analysis of Training Strategies and Uncovering of Network Decision *Frontiers in Human Neuroscience*
[34] Borra D, Fantozzi S and Magosso E 2020 Convolutional Neural Network for a P300 Brain-Computer Interface to Improve Social Attention in Autistic Spectrum Disorder *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1837–43
[35] Farahat A, Reichert C, Sweeney-Reed C and Hinrichs H 2019 Convolutional neural networks for decoding of covert attention focus and saliency maps for EEG feature visualization *Journal of Neural Engineering*
[36] Bressan G, Cisotto G, Müller-Putz G R and Wriessnegger S C 2021 Deep Learning-Based Classification of Fine Hand Movements from Low Frequency EEG *Future Internet* **13** 103
[37] Zhang C, Kim Y-K and Eskandarian A 2021 EEG-inception: an accurate and robust end-to-end neural network for EEG-based motor imagery classification *J. Neural Eng.* **18** 046014
[38] Zhao X, Zhang H, Zhu G, You F, Kuang S and Sun L 2019 A Multi-Branch 3D Convolutional Neural Network for EEG-Based Motor Imagery Classification *IEEE Trans. Neural Syst. Rehabil. Eng.* **27** 2164–77
[39] Roy A M 2022 An efficient multi-scale CNN model with intrinsic feature integration for motor imagery EEG subject classification in brain-machine interfaces *Biomedical Signal Processing and Control* **74** 103496
[40] Zhao D, Tang F, Si B and Feng X 2019 Learning joint space–time–frequency features for EEG decoding on small labeled data *Neural Networks* **114** 67–77
[41] Borra D and Magosso E 2021 Deep learning-based EEG analysis: investigating P3 ERP components *Journal of Integrative Neuroscience* **In press**
[42] Chollet F 2016 Xception: Deep Learning with Depthwise Separable Convolutions *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 1800–7
[43] Simonyan K, Vedaldi A and Zisserman A 2014 Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps *arXiv:1312.6034 [cs]*
[44] Vahid A, Mückschel M, Stober S, Stock A-K and Beste C 2020 Applying deep learning to single-trial EEG data provides evidence for complementary theories on action control *Commun Biol* **3** 112
[45] Kobler R, Sburlea A I and Müller-Putz G 2017 A comparison of ocular artifact removal methods for block design based electroencephalography experiments *Proceedings of the 7th Graz Brain-Computer Interface Conference 2017* ed G Müller-Putz, D Steyrl, S Wrissnegger and R Scherer (Verlag der Technischen Universität Graz) pp 236–41
[46] Ravanelli M and Bengio Y 2018 Speaker Recognition from Raw Waveform with SincNet *2018 IEEE Spoken Language Technology Workshop (SLT)* pp 1021–8
[47] Borra D, Fantozzi S and Magosso E 2020 EEG Motor Execution Decoding via Interpretable Sinc-Convolutional Neural Networks *XV Mediterranean Conference on Medical and Biological Engineering and Computing – MEDICON 2019* ed J Henriques, N Neves and P de Carvalho (Cham: Springer International Publishing) pp 1113–22
[48] Clevert D-A, Unterthiner T and Hochreiter S 2015 Fast and accurate deep network learning by exponential linear units (elus) *arXiv preprint*
[49] Srivastava N, Hinton G, Krizhevsky A, Sutskever I and Salakhutdinov R 2014 Dropout: a simple way to prevent neural networks from overfitting *The Journal of Machine Learning Research* **15** 1929–58

[50] Ioffe S and Szegedy C 2015 Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift *Proceedings of the 32nd International Conference on Machine Learning* Proceedings of Machine Learning Research vol 37, ed F Bach and D Blei (Lille, France: PMLR) pp 448–56

[51] Snoek J, Larochelle H and Adams R P 2012 Practical Bayesian Optimization of Machine Learning Algorithms *Advances in Neural Information Processing Systems 25* ed F Pereira, C J C Burges, L Bottou and K Q Weinberger (Curran Associates, Inc.) pp 2951–9

[52] Zhuang F, Qi Z, Duan K, Xi D, Zhu Y, Zhu H, Xiong H and He Q 2020 A Comprehensive Survey on Transfer Learning *arXiv:1911.02685 [cs, stat]*

[53] Kingma D P and Ba J 2017 Adam: A Method for Stochastic Optimization *arXiv:1412.6980 [cs]*

[54] Smith S and Nichols T 2009 Threshold-free cluster enhancement: Addressing problems of smoothing, threshold dependence and localisation in cluster inference *NeuroImage* **44** 83–98

[55] McInnes L, Healy J and Astels S 2017 hdbscan: Hierarchical density based clustering *JOSS* **2** 205

[56] Cecotti H and Graser A 2011 Convolutional Neural Networks for P300 Detection with Application to Brain-Computer Interfaces *IEEE Transactions on Pattern Analysis and Machine Intelligence* **33** 433–45

[57] Benjamini Y and Hochberg Y 1995 Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing *Journal of the Royal Statistical Society. Series B (Methodological)* **57** 289–300

[58] Yuan H, Perdoni C and He B 2010 Relationship between speed and EEG activity during imagined and executed hand movements *J. Neural Eng.* **7** 026001

[59] Filimon F, Nelson J D, Huang R-S and Sereno M I 2009 Multiple Parietal Reach Regions in Humans: Cortical Representations for Visual and Proprioceptive Feedback during On-Line Reaching *Journal of Neuroscience* **29** 2961–71

SECTION III: MOTOR DECODING FROM NEURONS' SPIKING RATE

CHAPTER 7: DESIGN OF A CNN FOR NEURONS' SPIKING RATE DECODING

The study reported in this chapter refers to the published journal paper entitled "Decoding sensorimotor information from superior parietal lobule of macaque via Convolutional Neural Networks" M. Filippini, D. Borra, M. Ursino, E. Magosso, P. Fattori, *Neural Networks*, 2022. In this study we adopted CNNs for neurons' spiking rate decoding. Both classification and regression problems were addressed, to provide a more complete evaluation of CNNs. In addition, the CNN structure was kept light, and the hyper-parameters were optimized via an automatic hyper-parameter search algorithm.

Despite the well-recognized role of the posterior parietal cortex (PPC) in processing sensory information to guide action, the differential encoding properties of this dynamic processing, as operated by different PPC brain areas, are scarcely known. Within the monkey's PPC, the superior parietal lobule hosts areas V6A, PEc, and PE included in the dorso-medial visual stream that is specialized in planning and guiding reaching movements. Here, a Convolutional Neural Network (CNN) approach is used to investigate how the information is processed in these areas. We trained two macaque monkeys to perform a delayed reaching task towards 9 positions (distributed on 3 different depth and direction levels) in the 3D peripersonal space. The activity of single cells was recorded from V6A, PEc, PE and fed to convolutional neural networks that were designed and trained to exploit the temporal structure of neuronal activation patterns, to decode the target positions reached by the monkey. Bayesian Optimization was used to define the main CNN hyper-parameters. In addition to discrete positions in space, we used the same network architecture to decode plausible reaching trajectories. We found that data from the most caudal V6A and PEc areas outperformed PE area in the spatial position decoding. In all areas, decoding accuracies started to increase at the time the target to reach was instructed to the monkey, and reached a plateau at movement onset. The results support a dynamic encoding of the different phases and properties of the reaching movement differentially distributed over a network of interconnected areas. This study highlights the usefulness of neurons' firing rate decoding via CNNs to improve our understanding of how sensorimotor information is encoded in PPC to perform reaching movements. The obtained results may have implications in the perspective of novel neuroprosthetic devices based on the decoding of these rich signals for faithfully carrying out patient's intentions.

7.1. INTRODUCTION

The posterior parietal cortex (PPC) houses several areas implicated in the integration of sensory stimuli (e.g., visual, somatosensory) to guide interaction with the surrounding environment [1,2]. Sensory information flows through different parietal areas, and different steps of integration support the movement control required for the interaction. A first functional subdivision was proposed by Ungerleider and Mishkin [3,4] with the division into ventral stream, from occipital to temporal lobe, the *what way*, and dorsal visual stream, to the parietal lobe, the *where way*. Subsequent division of the dorsal pathway into medial and lateral streams attributed the superior parietal lobule (SPL) to the branch of the medial dorsal stream, which host areas implicated in the spatial control of the action, against the areas of the inferior parietal lobule, more involved in the control of the properties of grasping [5,6]. In humans, lesions localized in the SPL lead to severe deficits in estimation and awareness of the spatial position of objects to be reached, such as those reported in optic ataxia [7,8].

Within PPC, V6A, PEc and PE are three contiguous areas located in the medial part of the SPL. V6A, the most caudal area of the SPL, hosts neurons modulated by different parameters linked to visuomotor coordination, including gaze signals [9–11], direction and amplitude of reaching [12,13] and spatial attention [14]. Rostrally to V6A, PEc maintains visual and reach related modulations but corroborated by increasing somatosensory inputs [15,16]. Finally, PE strongly respond to proprioceptive stimulation with a limited presence of visual information [17,18]. The segregated functional properties of these areas support the idea that caudal SPL (V6A) encodes for spatial position of targets, while rostral areas (PEc-PE) actively support the movement control and execution relying on prominent sensory feedbacks [15,16,19,20]. Given the latency in the feedback inputs, the system must implement an internal model of surrounding environment (and the consequences of actions performed in the environment) and expected feedbacks. Indeed, comparing expected with real feedback enables a much more powerful dynamical interaction with the environment. This model has been proposed to run in the PPC [21–23].

Despite the presence of numerous studies, a clear understanding of how these PPC's areas differentially encode visuomotor information to dynamically guide action is still lacking. In particular, these areas have been mostly characterized at single cell level [13,15,18] while, at the best of our knowledge, no work has comprehensively considered and directly compared the dynamics of information encoded, at the population level, in these three areas. Neural decoding, i.e., the use of activity recorded from multiple brain sources to predict variables in the external word, represents a useful tool to characterize how much information a given area contains about an external variable and how this information differs across different areas [24]. The attainment of different decoding performances when building separate decoders, e.g., each using neural activity from a different PPC area, may be indicative of a different amount of information encoded in each population.

Machine learning (ML) algorithms are widely used to design neural decoders [24]. Deep learning – a recently proposed branch of ML [25] – exploits models designed by stacking layers of artificial neurons, i.e., deep neural networks (DNNs). Remarkably, DNNs are capable to handle raw/lightly pre-processed neural time series as input, automatically learning during a training process the more relevant features to decode the brain states of interest while exploiting

all the information contained in the input signals. Therefore, DNNs represent an important advantage over more traditional ML approaches; the latter first extract and select features from input neural time series, and then finalize the decoding task, resulting in a workflow more driven by a priori knowledge about the expected underlying neural correlates. Furthermore, conversely to ML algorithms, DNNs provide a direct processing of the information from the neural signals to the desired output (in an end-to-end fashion), describing in general a complex non-linear function mapping input signals to desired outputs. Nevertheless, DNNs have some limitations, such as the introduction of many trainable parameters and the introduction of a second set of parameters, named hyper-parameters, that define the functional form of the decoder and must be set before the DNN training (e.g., the number of layers, the number of neurons per layer, etc.). Among DNNs, Recurrent Neural Networks (RNNs) were used to decode arm kinematics from spiking activity [26,27] also from PPC signals [28,29], generally using complex and heavy architectures (in terms of architecture structure and number of trainable parameters, respectively), with a fixed configuration selected via empirical evaluations (i.e., manual selection) or selected without any explicit criterion, and hardly interpretable. Therefore, these DNNs do not represent a parsimonious use of trainable parameters and, due to the high number of trainable parameters, DNNs could be more prone to overfit small datasets (commonly recorded in practice). Furthermore, the adoption of manual selection to define the DNN structure and assign hyper-parameters could lead to suboptimal decoding results, limiting the potentialities of DNNs, and require manual effort to be performed. Convolutional Neural Networks (CNNs) could also be used without hampering the decoding performance, reducing the number of trainable parameters with respect to RNNs and being easier to be interpreted in the learned features [30]. These algorithms are inspired by the hierarchical structure of the ventral stream of the visual system and thus, automatically learn hierarchical representations of the input signal with multiple levels of abstraction [31]. Despite being scarcely applied for the decoding of brain states directly from the neuron activity, there is a growing interest in the design and application of CNNs over other DNNs (such as RNNs), as reported in related fields of electroencephalogram decoding [32,33]. In addition, techniques aimed to automatically search the optimal hyper-parameter configuration of the decoder, such as Bayesian Optimization (BO), could be used to automatically design decoder functional forms without relying on mere empirical evaluations.

The aim of this study is to develop and use CNNs to accurately decode external variables (reaching goal and trajectory) from PPC neural activity. CNNs were used to catch temporal dynamics and model non-linearity distinctive of high-order areas. To this end, we recorded the activity of single neurons from macaque V6A, PEc, and PE areas while monkeys were performing a delayed reaching 3D task to 9 reaching targets. We approached two different decoding problems for a wider validation of CNNs. At first, we performed the classification of the 9 discrete spatial positions; this problem was addressed by predicting the output class as a function of time within the reaching trials, providing a dynamic decoding of the end-point during the reaching phases. To test whether the non-linear input/output mapping as performed by DNN methods was superior then simpler linear mapping, CNN classification performances were compared against a linear classifier. Then, a regression problem consisting in predicting the 3D hand trajectory of reaching was tackled; this problem was also useful to explore internal PPC model. Remarkably, to overcome the current limitation in designing and using DNNs for

neural decoding, the CNN structure and hyper-parameters were tuned using an automatic hyper-parameter search algorithm based on BO. With this approach, we mainly aspire:

i. To improve the state-of-the-art of decoding techniques. Indeed, at best of our knowledge, this is the first time that CNNs are validated and used to decode spiking neuron activity, and that Bayesian optimization is used in this context. This may represent a significant step forward, as CNNs may result highly lighter and faster than RNNs on one hand, and more accurate than simpler linear decoders on the other. The possibility to obtain decoders less handcrafted, more efficient, and accurate than other solutions may not only boost a better comprehension of the characteristics of information contained inside the decoded neural populations, but also have implications in neural engineering, such as helping advancement in brain-computer communication tools.

ii. To investigate how the reaching target position and the hand kinematics (3D position) are differently encoded in the three examined PPC areas at the population level, by analyzing the performance of the tested decoders across the three areas. Via this analysis, we wish to evidence how neural decoders, via data-driven input-output mapping, can have significant potential to inform about the nature of information contained in neural populations. We also expect that, when compared to the linear classifier, CNNs, taking into account nonlinearity and the temporal dynamics (via temporal convolutions) can better catch the characteristics of each area.

7.2. MATERIALS AND METHODS

The experimental part of this study was performed in accordance with the guidelines of the EU Directives (86/609/EEC; 2010/63/EU) and the Italian national law (D.L. 116–92, D.L. 26–2014) on the use of animals in scientific research. Protocols were approved by the Animal-Welfare Body of the University of Bologna. During training and recording sessions, particular attention was paid to any behavioral and clinical sign of pain or distress. More details on the experimental procedures can be found in [13,34–36].

7.2.1. Data acquisition

Electrophysiological recordings

Two male macaque monkeys (Macaca fascicularis) weighing 4.4 kg (Monkey 1, M1) and 3.8 kg (Monkey 2, M2) were used. Extracellular single-cell activity was recorded by means of single electrode from areas V6A, PEc and PE (Figure 7.1a). V6A is localized in the anterior bank of the parieto-occipital sulcus (POs) [35]. Next and rostrally to V6A, on the exposed surface of SPL, there is PEc. Finally, between somatosensory cortex and PEc, we recorded from the posteromedial part of PE [18]. Recording sites were assigned to V6A, PEc, and PE according to cytoarchitectural analyses of histological sections of the brain [35]. The procedure is based on the study of marking lesions and on other cues (e.g., coordinates of penetration within the recording chamber, distance of recording site from the surface of the hemisphere, etc.) [35]. We performed multiple electrode penetrations using a five-channel multielectrode recording system that permitted to record from up to five single electrodes at once (Thomas Recording GmbH, Giessen, Germany). We recorded the activity of 258 V6A neurons, 120 cells from M1 and 138 cells from M2, 214 neurons from PEc, 94 and 120 from M1 and M2 respectively, 141 from area PE, 71 and 70 from M1 and M2 respectively. Action potentials (spikes) in each channel were isolated with a waveform discriminator (Multi Spike Detector; Alpha Omega Engineering Nazareth, Israel) and were sampled at 100 kHz. The quality of the single-unit isolation was determined by the homogeneity of spike waveforms and clear refractory periods in ISI histograms during spike-sorting. Only well-isolated units not changing across tasks were considered.

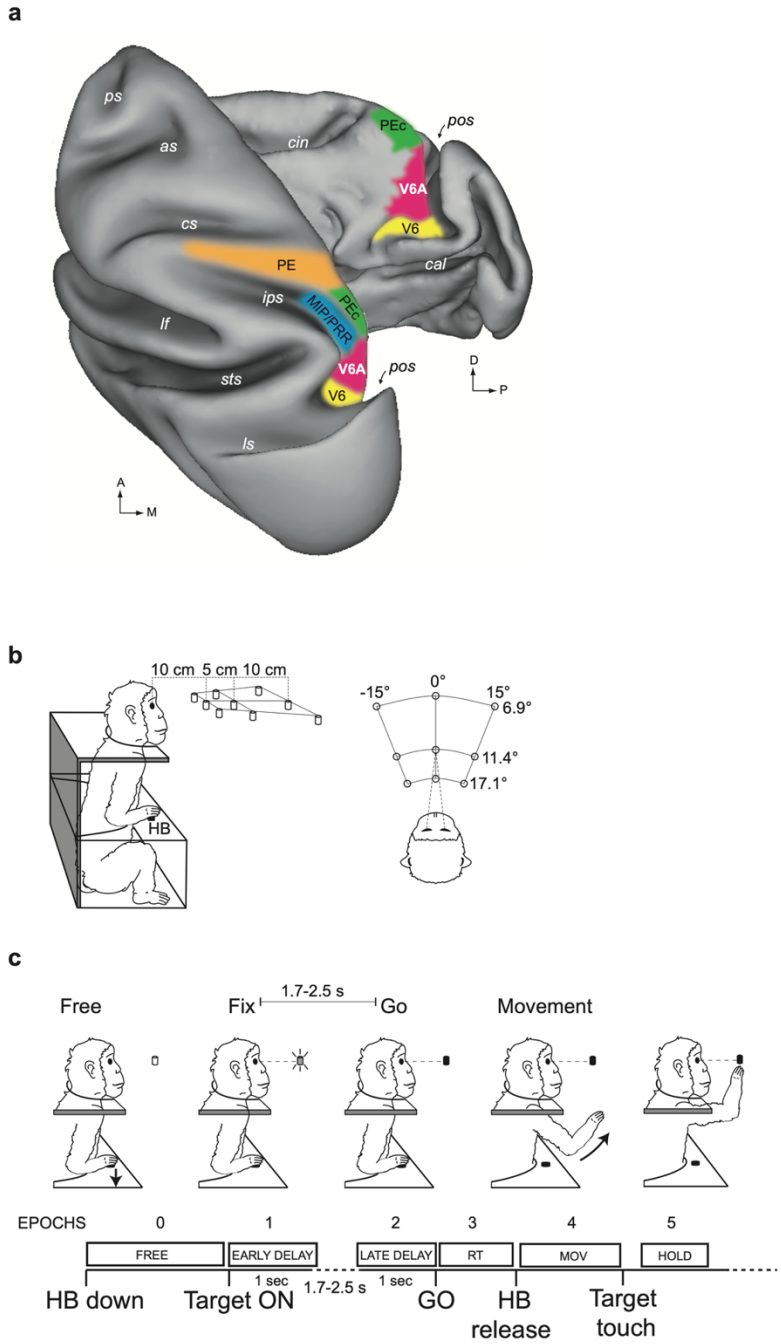


Figure 7.1 - Recording areas and Behavioral Task. (a) Dorsal view of left hemisphere (left) and medial view of right hemisphere (right) reconstructed in 3D. Caret software (http://brainvis. wustl.edu/wiki/index.php/Caret:Download) was used, showing the location and extent of V6A (magenta), PEc (green), PE (orange), MIP/PRR, medial intraparietal area/parietal reach region [38] (blue), and V6 [35] (yellow); cal, calcarine sulcus; cin, cingulate sulcus; cs, central sulcus; ips, intraparietal sulcus; lf, lateral fissure; ls, lunate sulcus; pos, parieto-occipital sulcus; ps, principal sulcus; sts, superior temporal sulcus; D, dorsal; P, posterior. (b) Schematic representation of the experimental setup for the reaching task. Exact distances are indicated in the lateral (left) and top (right) views. Nine LEDs are used as targets, embedded in a panel located at eye level. HB = home button. (c) Time courses and behavioral epochs in reaching tasks. The task starts with the animal holding down the HB (FREE 1sec, epoch 0), then a fixation LED lights up on one of the 9 positions, the animal starts to fixate and waits (first 1 sec of DELAY, EARLY DELAY, epoch 1) for the change of color of the LED that occurs at the end of the delay (LATE DELAY, last 1 sec of DELAY, epoch 2) with the GO event. The reaction time (epoch 3) is then from the GO signal to the actual release of the HB. The animal releases the HB to perform the

movement (MOV, epoch 4) then keeps the LED target pressed for the HOLD interval (epoch 5) to then return to the HB and begin the next trial.

Behavioral task

Monkeys sat in a primate chair (Crist instruments, Hagerstown, MD, USA) and were trained to perform a Fixation-to-Reach task under controlled conditions. This task was performed in darkness with the arm contralateral to the recording hemisphere. During the task, the animals maintained steady fixation on the cued (green) reaching target with their head restrained. Before starting the arm movement, the monkeys kept their hand on a button (home button [HB], 2.5 cm in diameter) located 5 cm in front of the chest on the midsagittal plane (Figure 7.1b left). Reaches were performed to one of nine light-emitting diodes (LEDs, 6 mm in diameter). The LEDs were mounted on a horizontal panel located in front of the animals, at different distances and directions with respect to the eyes but always at eye level, so the movement performed by the monkeys to reach and press the LED was upward. Target LEDs were arranged in three rows: one central, along the sagittal midline, and two laterals, at version angles of -15° and +15°, respectively (Figure 7.1b right). Along each row, three LEDs were located at vergence angles of 17.1°, 11.4°, and 6.9°. The nearest targets were located at 10 cm from the eyes, whereas the LEDs placed at intermediate and far positions were at a distance of 15 and 25 cm, respectively. The range of vergence angles was selected to include most of the peripersonal space in front of the animals, from very near (10 cm) to the farthest distances reachable by monkeys (25 cm).

The trial began when the animals pressed the button near their chest, outside the field of view (HB press). After 1s, one of the nine LEDs was switched on to green (Green-on). The monkeys had to fixate the LED within 500 ms, while keeping the HB button pressed. Then, the monkeys had to wait 1.7–2.5 s for a change in the color of the LED (from green to red) without performing any eye or arm movement. The latter color change was the go signal (Go) for the animals to release the home button and to start an arm movement toward the foveated target. Then, the monkeys reached the target and held their hand on the target for 0.8–1.2 s. When the target cue was switched off, the monkeys had to release this cue and return to the HB, which ended the trial and allowed the monkeys to receive a reward.

The task was performed in blocks of 90 randomized trials, including 10 repetitions for each target position (out of the 9 possible target positions). According to Figure 7.1c we divided each recording trial in 5 functional epochs. Epoch 0 represented the interval in which the monkey was not engaged in the task waiting for the LED turning on; Epoch 1 and Epoch 2 labelled the delay interval, specifically since the delay interval has random duration, it was separated in its first second (epoch 1) and last second (epoch 2); Epoch 3 represented the reaction interval; Epoch 4 was the movement interval (reaching movement towards the target point); Epoch 5 represented the holding interval (hold on the target point).

The luminance of LEDs was regulated to compensate for difference in retinal size between LEDs located at different distances. To prevent dark adaptation, the background light was switched on between blocks. The presentation of stimuli and the animals' performance were automatically controlled and monitored by LabVIEW-based software (National Instruments) as described previously in Kutz et al. [39], enabling the interruption of the trial if the monkey broke fixation, made an incorrect arm movement, or did not respect the temporal constraints

of the task described above. The correct performance of movements was monitored by pulses from microswitches (monopolar microswitches, RS Components, UK) mounted under the home button and each LED. At the beginning of each recording session, the monkeys were required to perform a calibration task to calibrate an eye tracker (ISCAN, see below). Calibration data was used to correct eye signals as they are dependent on the position of the cameras which can potentially change from day to day. For the calibration, animals sequentially fixated 5 LEDs mounted on a vertically arranged panel placed at a distance of 15 cm from the eyes. For each eye, we extracted signals for calibration during fixation of five LEDs, arranged in the shape of a cross. One LED was centrally aligned with the eye's straight-ahead position and four LEDs were peripherally placed at an angle of ±15° (distance: 4 cm) both in the horizontal and vertical axes. From the two individually-calibrated eye position signals, we derived the mean of the two eyes (version signal) and the difference between the two eyes (vergence signal) using the following equations: $version = (R + L)/2, vergence = R - L$, where R and L are the gaze direction of the right and left eye respectively, expressed in degrees of visual angle from the straight-ahead direction. Eye signals were monitored to be sure that the animal was staring at the target while performing the task, reducing possible modulation of neurons' firing rates due to saccade execution [40].

7.2.2. Data pre-processing and preliminary analysis

For each neuron and each individual recording trial, the activity was initially binned at 20 ms. Since the trials and epochs have a different duration, the use of a constant temporal window produces a different number of bins across trials, not allowing to construct a uniform dataset. Therefore, the average number of bins (across different neurons and trials) of each epoch was computed; then, the activity of each neuron and trial was re-binned by using that number of average bins per epoch. This procedure led to an activity binned slightly more or less with respect to the original 20 ms binning (20.1±1.9, mean ± standard deviation across monkeys and areas). Thus, to address this, we computed firing rates by dividing the number of spikes occurring within the bin by the temporal length of the bin. In the following, the neuron activity is described by means of its firing rate and to study the temporal dynamics of neural coding we constructed some short signals (named "chunks" in the following) composed of overlapped windows of B bins, extracted with stride of S bins to be used as inputs to the CNN.

As preliminary analysis, we looked for the neurons modulated by the reaching task. Only in this case, the neuron activity was divided into non-overlapped chunks of $B = 5$ bins (i.e., extracted with a stride of $S = 5$) bins and was analyzed using a sliding ANOVA to assess the variance in neuronal activity between the different conditions tested. One neuron was considered significantly modulated with $p < 0.01$. Results were plotted as number of modulated neurons over time.

7.2.3 CNN-based population decoding

Problem definition

Firing rates from every single neuron of the investigated population obtained from a specific monkey – identified with $m = \{"M1","M2"\}$ – and a specific recording area – identified with

$a = \{"V6A", "PEc", "PE"\}$ – were processed as follows to perform decoding. At first, overlapped chunks of shape (N, B) were extracted using a stride parameter S, where N denotes the number of neurons recorded for a given animal and area (variable across animals and areas, representing the spatial dimension, see Table 7.1 for N values across animals and areas), and B denotes the number of bins in each extracted chunk (representing the time dimension).

Table 7.1 – Number of recorded neurons for each animal and each recording area. In addition, the total number of training, validation and test examples in supervised problems 1 and 2 as resulting from the trial chunking procedure (see Section 7.2.2) are reported. Note that that the reported values for supervised problem 2 refer to the default assignment of the desired output (i.e., with no offset in the assignment).

			Supervised problem 1 (epochs 0-5)			**Supervised problem 2 (epochs 2-5)**		
Monkey	**Area**	**N**	**Training**	**Validation**	**Test**	**Training**	**Validation**	**Test**
M1	V6A	138	3312	414	2070	1872	234	1170
	PEc	120	3384	423	2079	1944	243	1179
	PE	70	3312	414	2061	1872	234	1161
M2	V6A	120	3312	414	2052	1872	234	1152
	PEc	94	3312	414	2061	1872	234	1161
	PE	71	3312	414	2043	1872	234	1143

By denoting with X_t the firing rates of each entire trial, and with $X_t[i]$ the i-th bin, chunks of neural activity were extracted as follows:

$$X_{t,i} = X_t[:, iS: iS + B - 1], 0 \leq\ i \leq M - 1, \tag{7.1}$$

indicating with $X_{t,i} \in \mathbb{R}^{N \times B}$ the i-th extracted chunk of firing rates for the t-th trial (see Figure 7.2a for a schematization of this chunking procedure), and with M the total number of chunks. $X_{t,i}$ represented the CNN input, while $y_{t,i}$ denotes the corresponding desired output. Thus, each so labelled dataset can be denoted by $D^{(m,a)}$:

$$D^{(m,a)} = \{(X_{t,0}, y_{t,0}), \dots, (X_{t,i}, y_{t,i}), \dots, (X_{t,M-1}, y_{t,M-1})\}^{(m,a)} \tag{7.2}$$

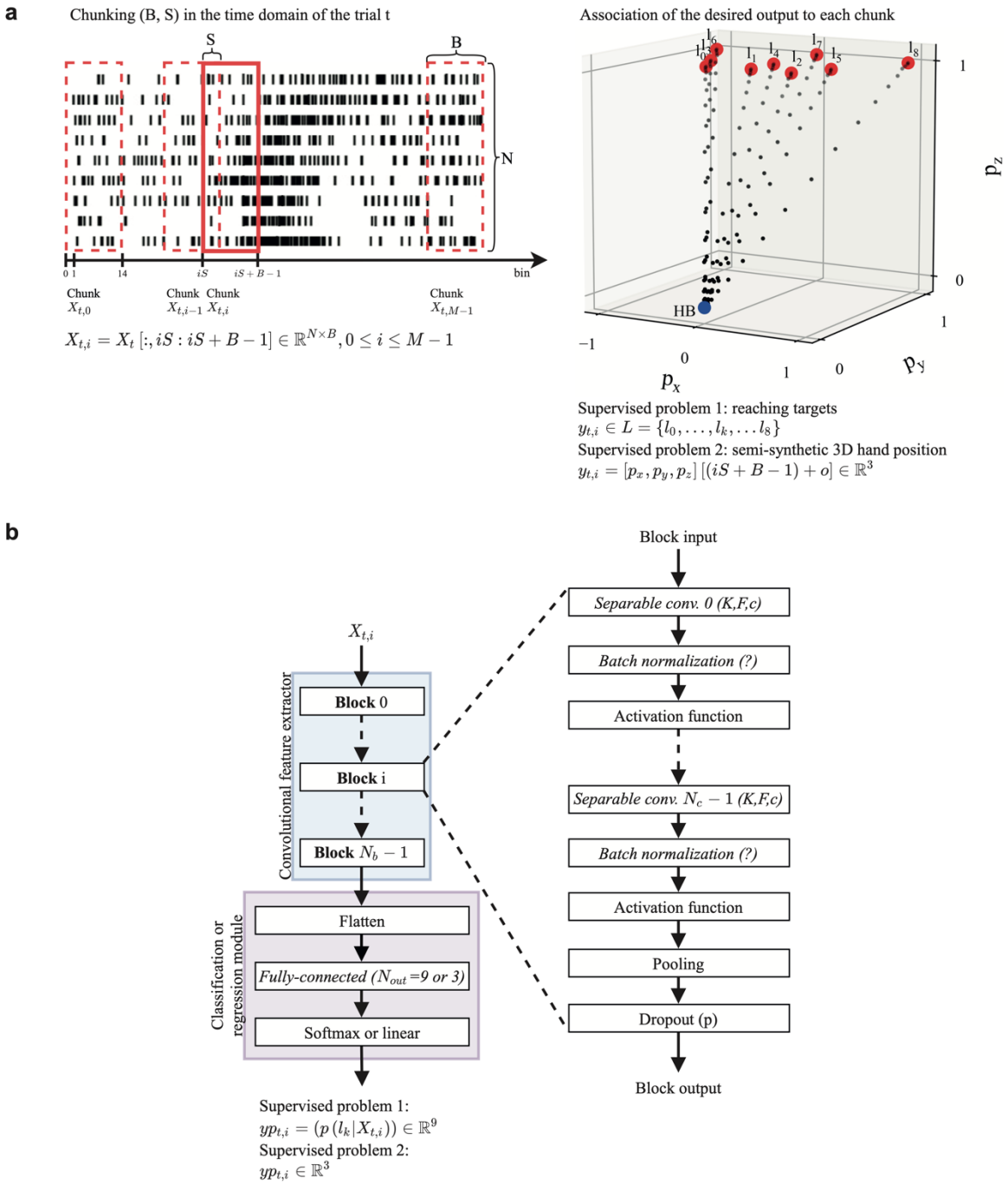


Figure 7.2 – Schematic representation of the dataset construction (a) and of the CNN design (b). (a) Neuron activities (left) and the desired output (right, corresponding to the reaching targets or hand trajectories) were processed to obtain $(X_{t,i}, y_{t,i})$, see Section 7.2.2. (b) $X_{t,i}$ is fed to the CNN, which provides $yp_{t,i}$ as output (predicted output). The CNN is composed by a convolutional feature extractor (blue box), followed by a classification or regressor module (purple box), see Section *CNN architecture* for the meaning of the symbols. Layers are represented by white boxes; layers introducing trainable parameters are denoted by the italic font, with the main associated hyper-parameters reported within brackets (see Section *CNN architecture* for the meaning of the symbols).

In this study, two different supervised problems were addressed using the same dataset. At first, we deepened the classification of targets to reach. In this case (supervised problem 1), $y_{t,i} \in L = \{l_0, \ldots, l_8\}$ was the desired class label assuming one among 9 possible values, corresponded to each target point that was constant in time during the trial (i.e., the end-point

to reach did not change within each trial). The supervision during classification was provided using one-hot encoded labels, i.e., forming an array corresponding to the true probability distribution with zeros for all classes except for the desired one which is 1. In addition, we investigated a regression problem consisting in the prediction of the 3D hand position while reaching targets (the same targets classified in the first supervised problem). In this case (supervised problem 2), $y_{t,i} \in \mathbb{R}^3 = [p_x, p_y, p_z]$ and corresponded to the 3D hand position coordinates at a specific time point to be predicted in the regression problem related to semi-synthetic trajectories (see Section *Supervised problem 2: Hand trajectory decoding* for further details). That is:

$$\begin{cases} y_{t,i} \in L = \{l_0, \dots, l_8\}, supervised\ problem\ 1 \\ y_{t,i} \in \mathbb{R}^3 = [p_x, p_y, p_z], supervised\ problem\ 2 \end{cases}. \quad (7.3)$$

Concerning the regression problem, as the hand position changed during each reaching trial, the desired output $y_{t,i}$ was assigned the 3D position coordinates in correspondence of the last bin of $X_{t,i}$ (referred as "default assignment", see Section *Supervised problem 2: Hand trajectory decoding* for further details), i.e., the prior neural activity of $B \cdot 20\ ms$ (e.g., 300 ms in case of $B = 15$ bins) was assigned to the current observable parameter, see Figure 7.2a for a schematization of this output association procedure. This corresponds to define $y_{t,i}$ as:

$$y_{t,i} = [p_x, p_y, p_z][(iS + B - 1) + o], \quad (7.4)$$

where $iS + B - 1$ denotes the time sample corresponding to the last bin of the chunk $X_{t,i}$, and o denotes an additional offset factor. The latter is set to 0 in the default assignment, while assumes positive or negative values in the latency analyses performed in Section *Supervised problem 2: Hand trajectory decoding*.

For both these supervised problems, the objective is to optimize - using a training set of examples (training stage) - a CNN described by a parametric model $f(X_{t,i}; \theta)$ parametrized in its trainable parameters θ, implementing a classifier in the first supervised problem, i.e., $f(X_{t,i}; \theta)$: $\mathbb{R}^{N \times B} \rightarrow L$, or a regressor in the second supervised problem, i.e., $f(X_{t,i}; \theta)$: $\mathbb{R}^{N \times B} \rightarrow \mathbb{R}^3$. As mentioned above, the CNN accepts as input the i-th chunk $X_{t,i}$ and provides the predicted output $yp_{t,i}$. This optimization corresponds to find the optimal $\theta^* = arg\ \min_{\theta} j(\theta)$, where $j(\theta)$ denotes the loss function which is computed based on the prediction error between the desired output $y_{t,i}$ and the predicted output $yp_{t,i}$. Afterwards, once the model is trained, it is tested on a separate test set (inference stage). Besides the trainable parameters contained in θ, the parameters defining the specific functional form of the decoder (e.g., number of convolutional layers, number of kernels to learn, etc.), called hyper-parameters, need to be set before training starts. However, the optimal hyper-parameter configuration is not a priori known and can be chosen either via an extensive empirical evaluation or via automatic hyper-parameter search. For both these solutions, to study the influence of different hyper-parameter configurations, the decoder needs to be validated on a separate set (different from the training and test set). Therefore, the dataset $D^{(m,a)}$ needs to be partitioned into separate training and test sets, respectively to optimize the parameters contained in θ and to evaluate the performance of the learning system on unseen data. In addition, to select the optimal hyper-parameter configuration (see Section *Hyper-parameter search via Bayesian Optimization (BO)*) a separate validation set needs to be designed. To perform such data partitioning, $D^{(m,a)}$ was

divided using a 10-fold cross validation scheme. Starting from 10 trials recorded for each spatial position (a total of 90 trials were available), each cross-validation fold included 72, 9, 9 trials in the training, test and validation sets, respectively, balancing across reaching targets (i.e., sets contained the same proportion among the 9 reaching targets). During the training stage, the parameters used to design $D^{(m,a)}$ were $B = 15$ and $S = 5$, while during the inference stage, these were $B = 15$ and $S = 1$. This choice allowed to train decoders with overlapped chunks of data i.e., augmenting the overall training set, but without increasing excessively the computational time ($S = 5$ during training), and to test the decoders on all possible chunks ($S = 1$ during testing). See Table 7.1 for the total number of training, validation and test examples, for each animal and recording area.

CNN architecture

The general structure of the CNN architecture is reported in Figure 7.2b and it is described in the following. The input layer of the CNN was represented by a 2D input feature map replicating in each neuron the corresponding value of the input example $X_{t,i} \in \mathbb{R}^{N \times B}$, i.e., the input layer was a 2D matrix with N rows and B columns representing the firing rates, of N neurons in B time bins. Afterwards, the input example was processed through a *convolutional feature extractor* to learn and extract relevant feature maps from the input example, followed by a *classification or regression module* that finalized the decoding depending on the addressed supervised problem addressed and based on the feature maps provided by the first module. Regarding the convolutional feature extractor, this was composed by stacking N_b convolutional blocks. Each convolutional block is composed by N_c repetitions of 1D temporal separable convolutional layers [41], each one learning K temporal kernels with a size of F, followed by batch normalization [42] (optional, depending on the hyper-parameter search) and a non-linear activation function. Then, after these N_c repetitions, each convolutional block included also a pooling layer – aimed to reduce the temporal dimension and thus, to reduce the overall model size by applying a pooling function (which is a hyper-parameter too, e.g., max or average pooling) – and a dropout layer [43], with dropout probability p. All convolutional layers were constrained (optional, depending on the hyper-parameter search) in their norm, keeping the norm of their parameters upper bounded at a constant c. Overall, batch normalization, dropout and kernel max norm constraints were introduced in the convolutional feature extractor to reduce overfit (i.e., regularization mechanisms). In addition, this module was designed using convolutional layers devoted to keep limited the number of trainable parameters, i.e., separable convolutions, preventing overfitting small datasets as the ones used in this study. The main hyper-parameters of the convolutional feature extractor were searched using an automatic hyper-parameter search algorithm (see Section *Hyper-parameter search via Bayesian Optimization (BO)* for further details).

The classification or regression module reshaped (flatten layer) at first the feature maps provided by the first module and included a fully-connected layer with N_{out} output artificial neurons to output the desired variables (target positions, $N_{out} = 9$ or hand position coordinates, $N_{out} = 3$). Depending on the addressed decoding problem, the activation function of the fully-connected layer changed. In case of classification, $N_{out} = 9$ neurons were activated using a softmax function, to provide as output the array $yp_{t,i} = p(l_k|X_{t,i}) \in \mathbb{R}^9, 0 \leq k \leq 8$ of the

predicted conditional probabilities associated to each target. Then, the most probable class was computed, i.e., $arg\max_{l_k} p(l_k|X_{t,i})$, and the decoded class obtained ($\in L = \{l_0, \dots, l_8\}$). In case of regression, $N_{out} = 3$ neurons were activated using a linear function, to directly provide as output the hand position coordinates $yp_{t,i} \in \mathbb{R}^3$while reaching targets.

Supervised problem 1: Target decoding

In the case of the classification problem, signals of the training, test and validation sets were standardized using the mean and standard deviation computed on the training set. The network used as input the signals pre-processed as described in Section 7.2.2 from epochs 0-5 (see Section *Behavioral task*) and provided as output the conditional probabilities for each target position. During the training stage, the loss function $j(\theta)$ was defined as the cross-entropy between the predicted distribution (provided by the CNN) and the empirical distribution (provided in the labeled dataset). During the inference stage on the test set, the CNN provided as output the probabilities that the input chunk belongs to each class; the predicted class was computed as the one with the highest probability among the 9 possible classes (see the possible reaching targets n Figure 7.2a right). Then, the decoding accuracy was computed based on the predicted and true classes. Accuracies on the test set were computed for each monkey and each recording area as a function of time, i.e., computing accuracies chunk by chunk. To provide a comparison with a state-of-the-art linear algorithm, a Naïve Bayes (NB) classifier as the one adopted in [44] was trained and evaluated with the same procedure adopted for the proposed CNN.

As the number of neurons recorded from the three areas differed (e.g., 70 neurons for PE vs. 100+ neurons in the other areas, see Table 7.1), we assessed whether differences in performance among different areas may be the consequence of a different number of neurons rather than intrinsically depend on differences in the information provided by neuron activities. To this aim, a dropping analysis was performed. During the d-th step of the dropping analysis, in each monkey and in each area, N_d neurons were randomly selected from $N_d = 2$ to $N_d = N$ with a step of 5 cells. That is, a subset containing N_d cells was randomly sampled from the original distribution and used to train, validate and test CNNs (using the same cross-validation scheme as described in Section *Problem definition*). In this way, the decoding performance was evaluated using the same number of N_d neurons in each area. In addition, this analysis simulates conditions of a reduced set of cells to decode, e.g., due to fibrosis around implanted electrodes, at different levels. The random sampling was performed 20 times for each d-th step of the dropping analysis and was performed for each monkey and each recording area. Due to the high computational cost of such simulation (involving >50K CNN optimizations), we applied the dropping analysis only for the supervised problem 1.

Supervised problem 2: Hand trajectory decoding

In the case of the regression problem, signals of the training, test and validation sets were standardized using the statistics computed on the training set, and the target coordinates were normalized between [-1,1] (centered on the mid sagittal axis of the animal), [0,1] (with 0 the position of the home button and 1 the maximum distance from the body corresponding to the

farthest target), and [0,1] (with 0 the elevation of the home button and 1 the height of the panel at eye level), respectively for the x-, y-, z-axis . To decode the hand trajectory, the network used as input the signals pre-processed as described in Section 7.2.2 from the epochs 2-5 (see Section *Behavioral task*) and provided as output the x, y, z hand position coordinates during the reaching of each target position.

Kinematic data of hand trajectories was not available for recorded neurons, therefore the reference trajectory of the hand during the experiment was reconstructed semi-synthetically, modelling the movement as a straight ballistic motion from the button near the chest to one of 9 positions on the panel (see the trajectories in in Figure 7.2a right) and imposing a classic bell-shaped profile for the acceleration and deceleration [45]. The bell-shaped profile (a Gaussian bell) was fitted to the average speed profile collected from real kinematic data, calculated over 144 trials on a separate monkey executing the same 3D reaching task, using as reference the x, y, z position of index finger. A motion capture system (VICON 460, 100 Hz sampling rate) recorded the 3D position of a reflective marker placed on the monkey's index finger. Data were run through a fifth-order Butterworth low-pass filter, finally trajectory were downscaled to 0-100% of movement to make possible aligning the different trials collected. Mean and standard deviation of the Gaussian bell determined the peak of maximum velocity (43% of movement time) and acceleration/deceleration ramp (σ=18% of movement time) and were used to reconstruct the reference hand trajectory for all cells.

As here we were interested in epochs including the last part of the waiting period until the target was maintained, the CNN had to learn to hold the initial position during the waiting time (0,0,0), gradually move toward the target during the movement interval, and hold the position during the last interval. During the training stage, the loss function $j(\theta)$ was defined as the mean squared error between the predicted trajectory value (provided by the CNN) and the empirical trajectory value (provided in the labeled dataset). During the inference stage on the test set, the network output was directly the predicted trajectory value for the corresponding input firing rate chunk. The predicted trajectory was then obtained by rearranging all values in the time-domain, and was compared with the semi-synthetically reconstructed trajectory, used as a ground truth. R-squared values were computed for each monkey and each recording area (reporting mean and standard deviation across folds).

In addition, only for the supervised problem 2, we conducted an analysis to study possible latencies, i.e., -120 ms, -40 ms, 40ms, 120 ms, between neuron activity (firing rates contained in each chunk $X_{t,i}$) and detected behavior (instantaneous trajectory value in its x, y, z position coordinates). To this aim, we introduced a time shift o between the desired output (x, y, z position coordinates) assignment and the neuron activity when designing the datasets $D^{(m,a)}$ and we trained, validated and tested CNNs for each offset condition. By default (see Section *Problem definition*), a zero offset ($o = 0$) was used, indicating that the assigned 3D trajectory value was sampled in correspondence of the last bin of the input chunk (i.e., to the 15th bin, see Section *Problem definition*). In addition to the default assignment, we deepened other conditions, by using offsets $o = \{-6, -2, 2, 6\}$. Positive (or negative) offset values denote conditions where CNNs were forced to learn features from past (or future) neuron activity ($X_{t,i}$). This suggests that neurons are coding for future trajectories (feedforward anticipation) or past trajectories (sensory feedback). As an example, when $o = +2$, position coordinates

sampled at +40 ms($= o \cdot 20\ ms$, see Section 7.2.2) in the future respect to input neuron activity were decoded.

Hyper-parameter search via Bayesian Optimization (BO)

Deep learning-based algorithms are defined by many hyper-parameters that are not a priori known. Therefore, in this study, to identify the optimal configuration of the convolutional feature extractor, automatic hyper-parameter search via BO [46] was adopted. This algorithm was applied to the identification of the optimal hyper-parameter set in the supervised problem 1. Due to the performed data split (10-fold cross-validation scheme) and to the nature of the dataset (2 monkeys and 3 recording areas), BO led to an optimal configuration specific for each fold, monkey, and area (60 configurations in total). Once BO was performed, the most frequent value (across folds, monkeys and areas) of each hyper-parameter was computed and used to design the CNNs to be trained from scratch in both supervised problems 1 and 2. Therefore, this BO-based procedure was used to identify a single configuration of hyper-parameters occurring more frequently across folds, monkeys and areas, i.e., a single functional form of the decoder f that was then exploited to train CNNs in both supervised problems. CNNs were trained – within each BO iteration and while training the most frequent CNN configuration in the supervised problems 1 and 2 – using Adam optimizer [47] with a batch size of 64 for a maximum number of 1000 epochs and applying early stopping on the validation loss.

In the following, an overview of automatic hyper-parameter search and of BO is reported. Hyper-parameter optimization is devoted to find the hyper-parameter configuration of a learning system (e.g., a CNN) associated with the best performance measured on a separate validation set. Let us denote with h the array containing the hyper-parameters of interest, with $h \in H$ where H is the hyper-parameter search space. In this study, we investigated the main hyper-parameters defining the convolutional feature extractor: N_b (number of blocks), N_c (number of separable convolutional layers per block), K (number of filters per layer), F (filter size), c (max norm constraint), the use of batch normalization, the activation function for the convolutional layers, the pooling function, the dropout probability p and the learning rate. Formally, hyper-parameter optimization consists in finding $h^* = arg\min_{h \in H} k(h)$, where $k(h): H \to \mathbb{R}$ represents the objective function to be minimized on the validation set (the loss function in this study). To evaluate $k(h)$ for each configuration h the learning system needs to be trained on the training set and then evaluated on the validation set.

Depending on the model complexity (typically high for deep learning-based decoders) and on the number of hyper-parameters to optimize, the evaluation of a trained model on the validation set can be expensive. Common hyper-parameter search algorithms (e.g., grid search or random search), perform many evaluations on the validation set, each one using a trained model with a hyper-parameter configuration based on a pre-defined rule (e.g., by sampling all possible hyper-parameter configurations or by randomly sampling a fixed number of configurations) ignoring the results of past evaluations. This often leads to wasting time in evaluating 'bad' hyperparameters. Bayesian optimization methods overcome this limitation, as they suggest in an informed way the next hyper-parameter configuration to be evaluated, thus, investigating hyper-parameters that seem promising based on past evaluations. Specifically, these methods build a Bayesian statistical model $p(k|h)$ of the objective function, called

surrogate probability model, which maps hyper-parameter values to the probability of getting a certain value of the objective function. The surrogate model is formed by keeping track of the past evaluation results and is easier to optimize than the actual objective function $k(h)$; thus, the next set of hyper-parameters to be evaluated on the actual objective function is chosen by selecting the hyper-parameters that perform best on the current surrogate model. Once the surrogate $p(k|h)$ has been initialized, the procedure involves several optimization iterations (100 iterations were performed in this study), run sequentially one after another, with each iteration consisting of the following steps:

i. Optimize the surrogate finding the hyper-parameters that perform best on the surrogate. The criterion used to optimize the surrogate is called "selection function".
ii. Design the learning system using the hyper-parameters selected at point i. Train the learning system and evaluate the objective function k.
iii. Update the surrogate probability model depending on the history of past evaluations, including the last evaluation result (at point ii).

Different choices exist for the surrogate probability model and criterion function used to optimize it. In this study, as commonly adopted [48], Tree Parzen Estimator (TPE) and Expected Improvement (EI) were used as surrogate model and selection function, respectively. By applying the Bayes rule, the surrogate probability model can be expressed as $p(k|h) = p(h|k)p(k)/p(h)$, and by using TPE, $p(h|k)$ is modelled as:

$$p(h|k) = \begin{cases} l(h), k < k^* \\ g(h), k \geq k^*, \end{cases} \tag{7.5}$$

where $l(h)$ and$g(h)$ are the distributions of the hyper-parameters, one modelled by using the previously evaluated hyper-parameters that resulted in objective function below the threshold k^*, and the other by using the previously evaluated hyper-parameters that resulted in objective function above the threshold k^*. These distributions are modelled with Gaussian mixture models in TPE. To initialize the algorithm (i.e., initialize the values needed to model the distributions) 20 iterations were performed by randomly sampling the hyper-parameters (i.e., performing random search). Furthermore, the TPE algorithm depends on the threshold k^*, the latter is chosen larger than the lowest observed k so that some points can be used to model $l(h)$. The algorithm selects k^* so that $\gamma = p(k < k^*)$, but no specific modelling for $p(k)$ is needed [48]. In this study, hyper-parameter values were all sampled from uniform distributions (defining $p(h)$) over the values reported in Table 7.2. Then, the expected improvement (EI, expectation that the surrogate model, by using h, assumes values below the threshold k^*) can be computed as:

$$EI_{k^*}(h) = \int_{-\infty}^{\infty} \max(k^* - k, 0)\, p(k|h)\, dk = \int_{-\infty}^{k^*} (k^* - k)\, p(k|h)\, dk. \tag{7.6}$$

In this scenario, the optimization problem (point i.) is reduced to a maximization of the EI. As reported in [48], by expressing $p(k|h)$ using the TPE modelling (Equation 7.5):

$$EI_{k^*}(h) \propto \left(\gamma + (g(h)/l(h))(1-\gamma)\right)^{-1} \propto l(h)/g(h). \tag{7.7}$$

Therefore, maximizing $EI_{k^*}(h)$ corresponds to maximizing the ratio $l(h)/g(h)$, i.e., find the optimal h with high probability under $l(h)$ and low probability under $g(h)$. Then, the true objective function $k(h)$ is evaluated with this optimal h (point ii.) and, subsequently, the two distributions $l(h)$ and $g(h)$ defining $p(h|k)$ are updated depending on the history of the past evaluations by taking into account the result of this last iterations (point iii.).

Table 7.2– Hyper-parameter space of the convolutional feature extractor. N_b and N_c denote the number of convolutional blocks and temporal separable convolutions per block, respectively. K and F denote the number of temporal kernels and the kernel size, respectively. Lastly, c and p indicate the maximum norm to use in max-norm constraint and the dropout probability, respectively. The hyper-parameter values were sampled using uniform distributions during the hyper-parameter optimization. Among the values, "None" denotes no usage of a specific technique (i.e., no use of kernel max norm constraint and no use of dropout).

Hyper-parameter	**Values**
N_b	[1, 2]
N_c	[1, 2, 3, 4]
K	[4, 8, 16, 32]
F	[3, 5]
c	[None, 0.25, 0.5, 0.75, 1]
Use batch norm.	[False, True]
Activation function	[ReLU, ELU [49]]
Pool function	[max, avg]
p	[None, 0.25, 0.5]
Learning rate	[0.0001, 0.0005, 0.001]

7.3. RESULTS

We recorded the activity of single neurons from 3 contiguous areas, V6A, PEc, and PE, in the superior parietal lobule of 2 macaques. In this study we were interested in testing if these three areas encode spatial information about the reaching goal and reaching trajectories with the same strength, and if they encode the temporal dynamics of this encoding. CNNs were used as decoders for the addressed supervised problems, and we were also interested in searching an optimal CNN design via automatic hyper-parameter using BO. In the following, the main results are reported.

7.3.1. Preliminary data analysis

A simple sliding ANOVA (Figure 7.3) was enough to show that the percentage of neurons modulated by spatial position of reaching targets was different depending on the considered area. PE showed half of neurons modulated (with peaks of approx. 20%) compared to V6A and PEc (with peaks between 50-60%) which were very similar. It is also interesting to note that the percentage of modulated cells was not stable over time but roughly was characterized by 2 prominent peaks in all areas: a first increase of modulated cells in the first phase of target presentation (Epoch 1), a second peak during the execution of the reaching movement (Epoch 4). These differences prompted us to decode the overall dynamics, which also reflect non-linear interactions and temporal aspects. Such dynamics can be shown with deep learning-based decoders such as CNNs and are reported in the following sections.

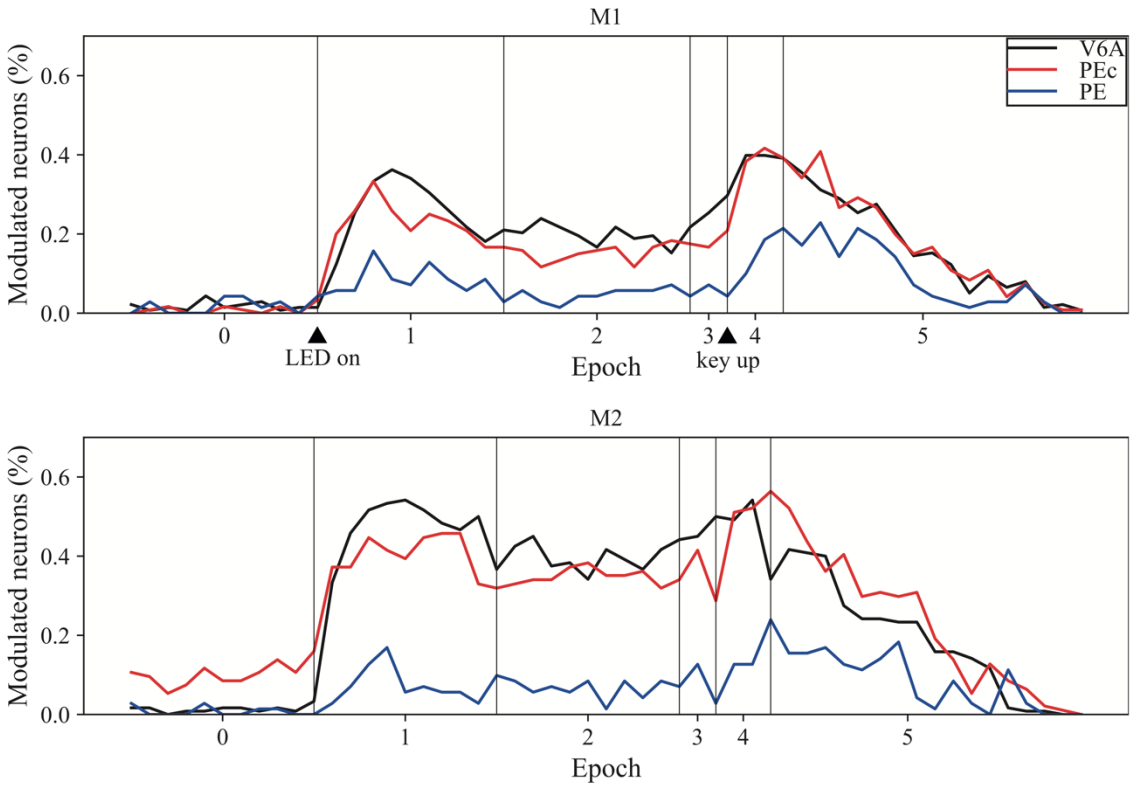


Figure 7.3 - Percentage of modulated neurons as obtained with a sliding ANOVA. The modulation is reported over time for each monkey (M1 on top and M2 on bottom) and each recording area (V6A black, Pec red, PE blue). Vertical bars denote the separation between epochs 0-5.

7.3.2. Optimal convolutional feature extractor

The convolutional feature extractor of the adopted CNN underwent automatic hyper-parameter search via BO in the supervised problem 1. Convolutional filters slide over the temporal dimension catching temporal patterns in firing rates; each training sample included 15 temporally consecutive bins, each bin represented the firing rate of a time interval of 20ms (see methods for more details). In Figure 7.4 the distributions of the searched hyper-parameters are reported in cumulative histograms, considering all the optimal configurations across folds, monkeys, and areas (60 configurations in total). From these results the most frequent configuration (higher bar on each plot corresponding to a given hyper-parameter in Figure 7.4) was a simple shallow CNN, characterized by a number of blocks $N_b = 1$, number of convolutional layers per block $N_c = 1$, number of convolutional filters $K = 32$, filter size $F = 5$, and max norm $c = 1$. Furthermore, options such as no batch normalization, ReLU activation functions for hidden units, and average pooling were more frequently adopted. Lastly, a dropout probability $p = 0.5$ and a learning rate of 0.001 were optimal. The CNN defined by this specific hyper-parameter configuration was used to solve the decoding problems 1 and 2, and the subsequent reported results are related to this specific functional form of the decoder.

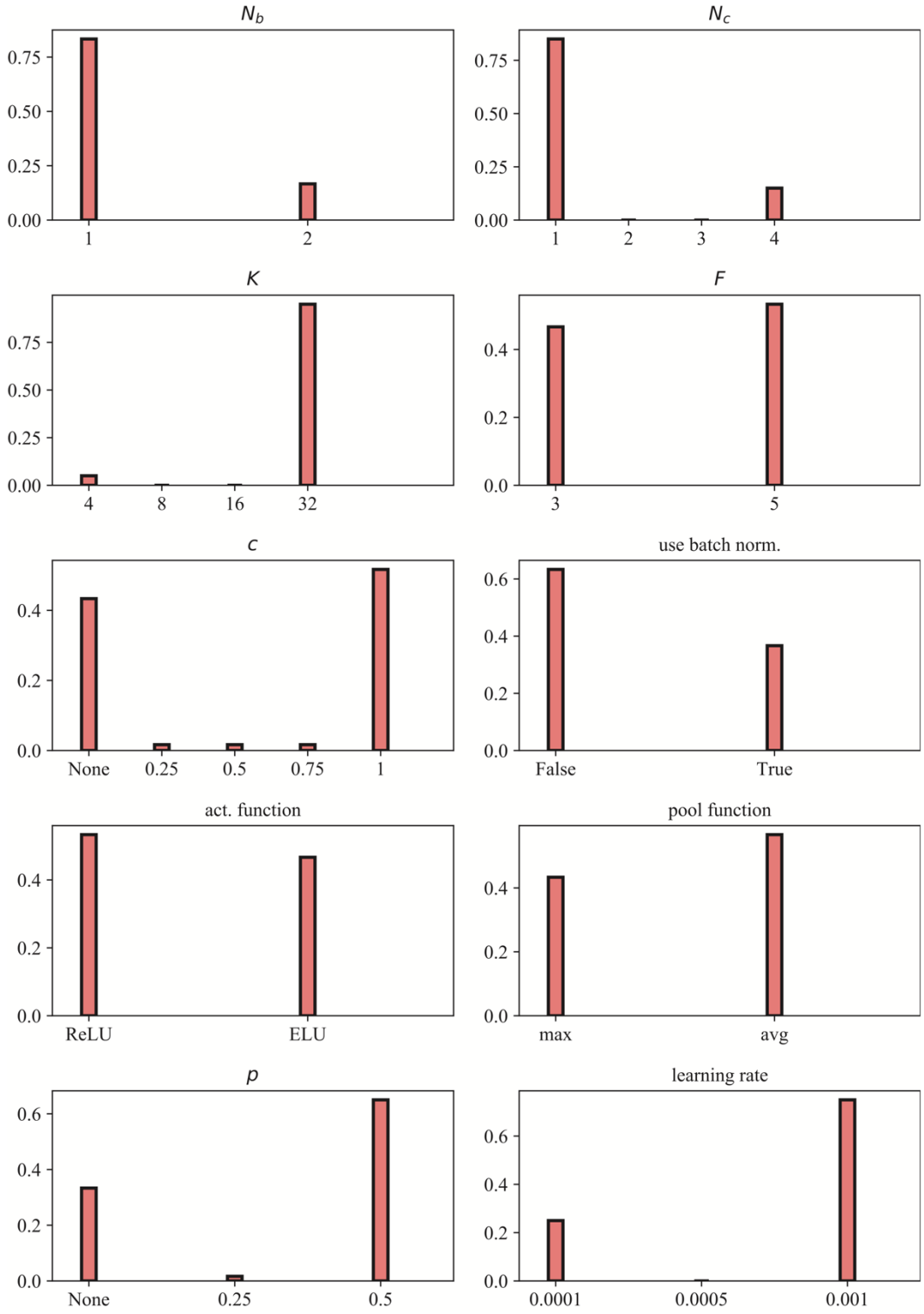


Figure 7.4 - Hyper-parameter probability distributions of the convolutional feature extractor resulting from BO. On the x-axis the hyper-parameter values are reported, while on the y-axis the probability that BO selected as optimal each hyper-parameter value is reported. N_b and N_c denote the number of convolutional blocks and temporal separable convolutions per block, respectively. K and F denote the number of temporal kernels and the kernel size, respectively. Lastly, c and p indicate the maximum norm to use in max-norm constraint and the dropout probability, respectively.

7.3.3. Supervised problem 1: Target decoding

In Figure 7.5 decoding accuracies are reported as a function of time, for both monkeys and all areas. CNNs learned to accurately map the activity of the collected neurons to the spatial location of the targets, as demonstrated by the average accuracy well above the chance level (11%) in both monkeys and in each area.

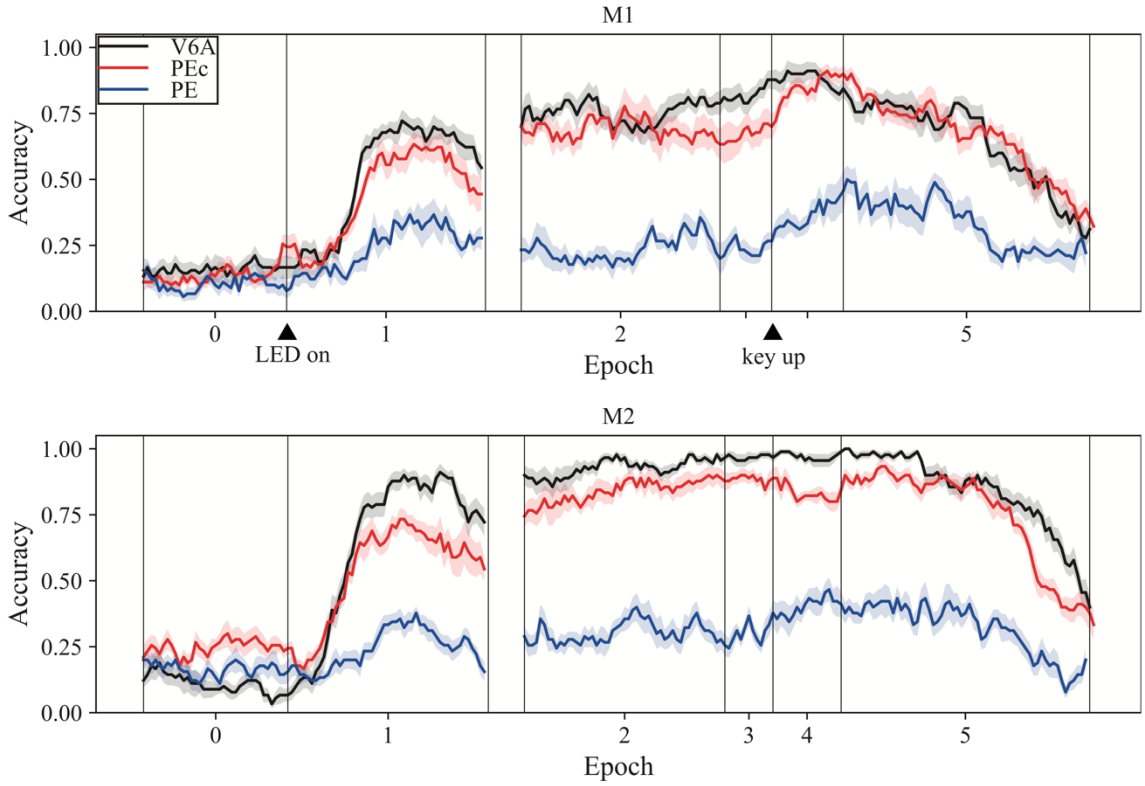


Figure 7.5 - Decoding accuracy over time. The performance metric obtained in the supervised problem 1 for each monkey (M1 on top and M2 on bottom) and each recording area (V6A, PEc, PE) used in the decoding is reported as a function of time. Mean values are reported (thick lines) ± standard error of the mean (overlayed shaded areas) across folds. Vertical bars denote the separation between epochs 0-5.

Accuracies began to increase with target detection (epoch 1), remained sustained with a ramping trend during movement preparation (epoch 2), peaked during movement execution (epoch 4), and then began to decline as touch on the target was maintained (epoch 5). In the case of V6A and PEc a maximum accuracy in decoding the correct target position above 80% was reached in epoch 4. Although the trend was similar for the 3 areas, PE had significantly lower accuracies than V6A and PEc, varying between 20-40%. This lower accuracy of decoding in area PE could be due to the lower percentage of modulated cells in PE within each epoch (see Figure 7.3) and/or to the smaller population available (roughly 1/3 of neurons were available for PE with respect of V6A and PEc, see Section *Electrophysiological recordings*). To better explore this last point, a dropping analysis (see Section *Supervised problem 1: Target decoding*) was applied, and its results are reported in Figure 7.6.

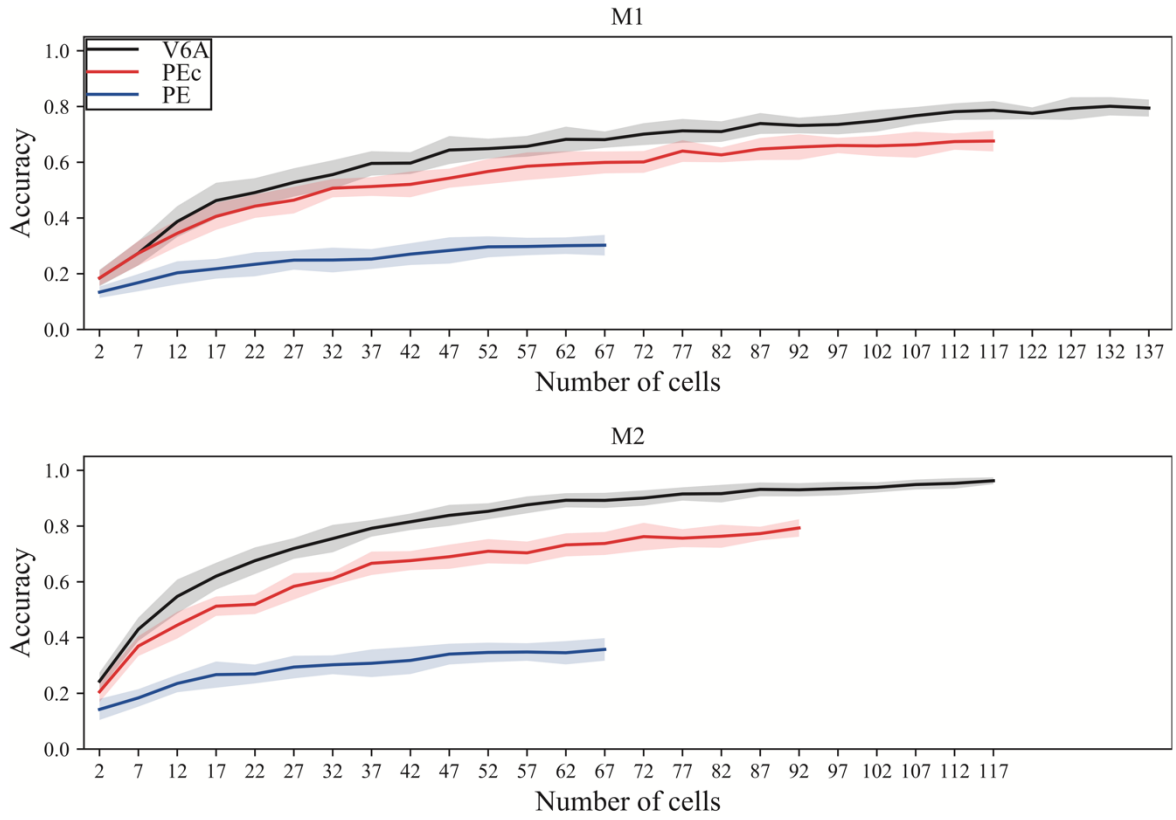


Figure 7.6 - Neuron dropping analysis. The decoding accuracy in the supervised problem 1 for each monkey (M1 on top and M2 on bottom) and each recording area (V6A, PEc, PE) used in the decoding is reported as a function of the number of cells used to classify reaching targets. For each step of the dropping analysis, the performance was averaged across folds. The figure reports the mean values (thick lines) ± standard error of the mean (overlayed shaded areas) across the 20 random samplings.

Here, accuracies as a function of the number of cells (N_d) used to decode the reaching targets are reported for each monkey and each recording area. For V6A and PEc, few neurons were enough to obtain accuracies well above the chance level, e.g., from 7 sampled neurons accuracies >30% were achieved both in monkey 1 and 2 (M1 and M2). Furthermore, the initial slope of the curve 'accuracy vs. number of cells' followed V6A>PEc>PE. Lastly, in V6A and PEc the trend kept improving more than PE as the number of neurons available increased. The dropping analysis therefore confirmed a lower ability to decode spatial information from area PE neurons compared to the better performance on V6A and PEc. These last two areas appear very similar in their ability to encode the information of the target position with the same strength.

Finally, Figure 7.7 shows the comparison between the decoding accuracy as a function of time obtained with the CNN-based decoder and the NB-based decoder.

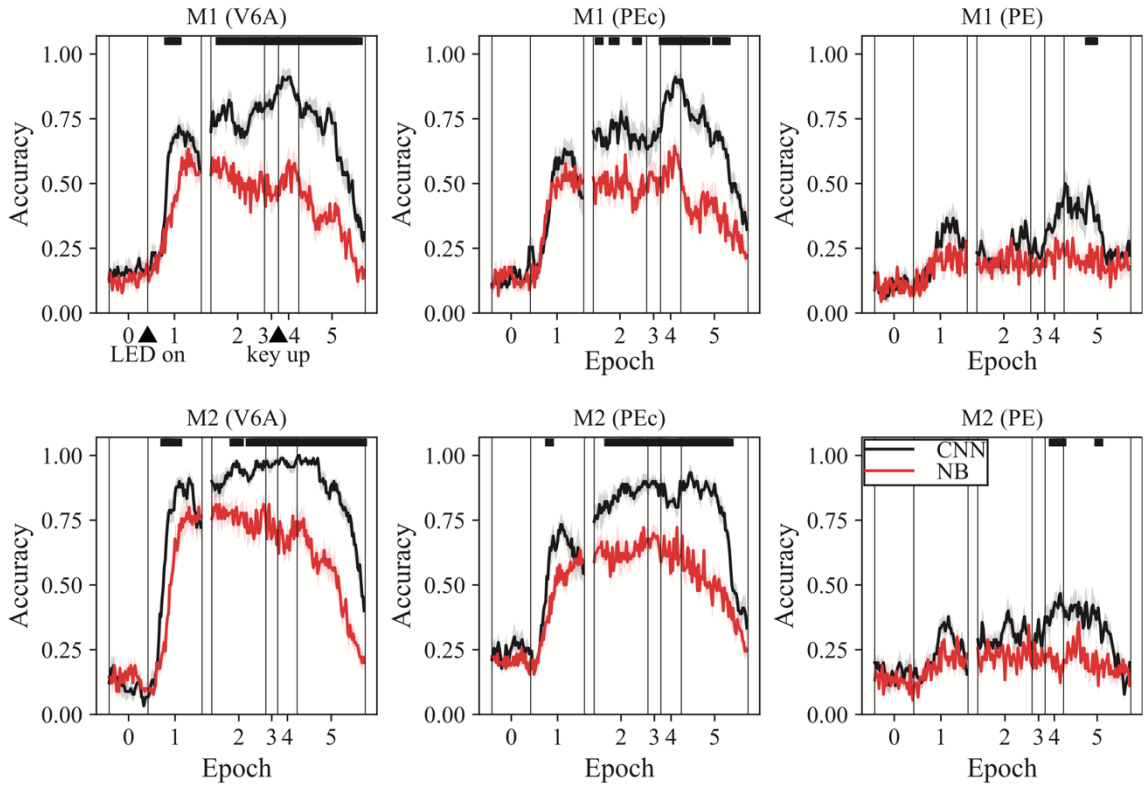


Figure 7.7 - Decoding accuracy over time obtained with the proposed CNN and with a NB linear classifier. The accuracy scored with both the CNN (black) and NB (red) in the supervised problem 1, for each monkey (M1 and M2) and each recording area (V6A, PEc, PE), is reported as a function of time. The figure reports the mean values (thick lines) ± standard error of the mean (overlayed shaded areas) across folds. Vertical bars denote the separation between epochs 0-5. Permutation cluster t tests were performed for each monkey and each recording area to analyze differences between the two algorithms; temporal intervals with significant performance differences ($p < 0.05$) between the two algorithms are reported on top of each panel with thick horizontal bars.

Overall, our proposed decoder exhibited higher accuracy scores. Moreover and interestingly, the time pattern of accuracy during the reaching movement differed across the two classifiers. The NB classifier after the initial increase in epoch 1, tended to exhibit an about constant (or slightly decreasing) accuracy across the other epochs, declining in epoch 5, and did not show the increasing trend peaking in the movement and hold phases as the CNN classifier. Statistical analysis shows that CNN outperformed ($p<0.05$) the NB-based decoder for all monkeys and areas, especially after the movement onset (e.g., during epoch 4 and 5), with improvements up to 46% (in M2 decoding from V6A, during epoch 5). These differences may arise from the capability of the CNN to learn non-linear dependencies exploiting complex hierarchical features in the temporal domain from the input temporal samples, while the classic linear algorithm based on NB is plausibly unable to catch these dynamics as it linearly combines the inputs and assumes conditional independences between the input temporal samples.

In this problem, we reconstructed discrete spatial positions, targets of the reaching. The 9 positions (9 classes) were recognized by CNN with high accuracy during all time intervals in which the animal was aware of the reaching position, from the first stages of target fixation to the holding of touch on the target. Besides decoding target position, we also tested whether and

to what extent other movement aspects, such as hand trajectories, could be potentially decoded from the activity of neurons in the investigated PPC areas.

7.3.4. Supervised problem 2: Hand trajectory decoding

A representative result while decoding position coordinates using V6A signals is reported in Figure 7.8a (with an offset=0 in the kinematic association, see Section *Supervised problem 2: Hand trajectory decoding*).

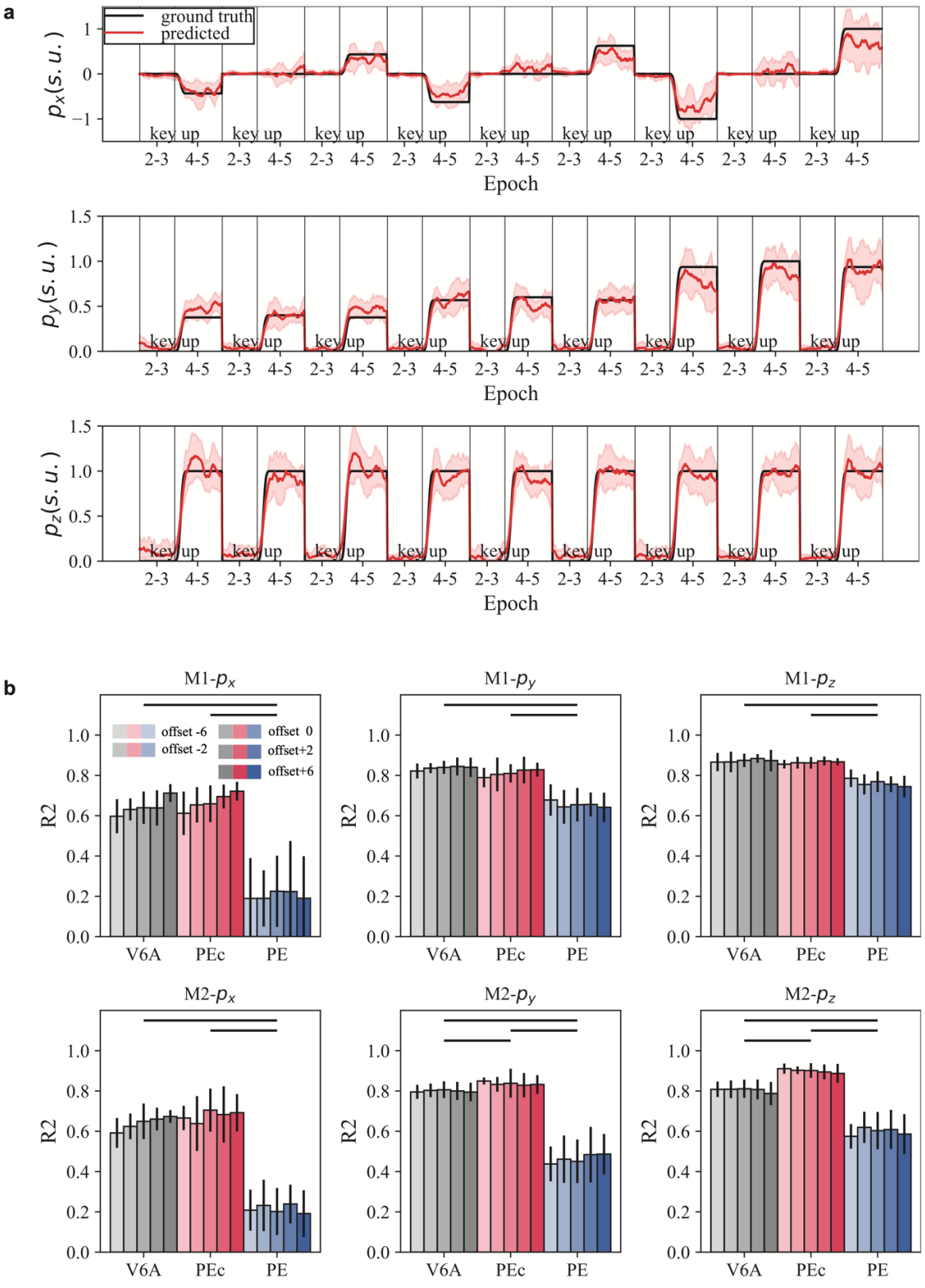


Figure 7.8 - Representative example of 3-D decoded trajectory (a) and R2 regression scores (b). (a) A representative example of decoded trajectory (red) alongside with the ground truth (black) over time (epochs 2-5) using V6A neural activity from M2 is reported. Coordinates are reported in their scaled units (s.u., see Section *Supervised problem 2: Hand trajectory decoding*). Mean and standard deviations (shaded area) are calculated over cross validations. (b) The performance metric obtained in the supervised problem 2 while decoding position

coordinates is reported for each monkey (M1 on top and M2 on bottom) and recording areas (V6A, PEc, PE) used in the decoding. Each plot reports R2 also as a function of the offset chosen while associating the target label (see Section *Supervised problem 2: Hand trajectory decoding*), where the default association is denoted by offset=0. R2 scores are reported in their mean values (bar height) ± standard deviation (vertical line) across folds. Horizontal black bars connect the areas found with significantly different R2.

Epochs 2-5 for the 9 targets are concatenated one after the other to show the range of x, y and z along the whole movement executed by the monkey. Predictions are plotted against ground truth trajectories. In this example, it is clear how V6A strongly contains enough information to accurately decode movement 3-D trajectory. However, not all areas allowed to reconstruct the movement trajectory with the same fidelity. In agreement with previous results as to target position decoding, area PE contains less useful information for the reconstruction of the trajectory, as shown by Figure 7.8b reporting the CNN performance metric for each monkey and recording area. Decoding performance using the default kinematic assignment (i.e., $o = 0$, see Section *Supervised problem 2: Hand trajectory decoding*) of the x-coordinate in V6A were significantly lower (for both monkeys, $p<0.01$, Wilcoxon signed-rank test corrected for multiple tests using the Benjamini-Hochberg procedure [50]) than y- and z-coordinates, but no differences were found between y- and z-coordinates in the same area. This may be multifactorial, in part given by a methodological drawback, the greater spread of the x (between -1 and 1) with respect to y and z (both between 0 and 1), but has also physiological substantiation. Indeed, previous studies (see Section 7.4.2) have demonstrated a prominent encoding of the depth component in areas of the superior parietal lobule, and this could be the cause of a greater difficulty in reconstructing the x component of trajectories.

In Figure 7.8b, decoding performance were reported also as a function of the offset used in the multi-lag kinematic association adopted in this study, i.e., ±6, ±2, 0 (see Section *Supervised problem 2: Hand trajectory decoding*). This has allowed us to probe whether feedback or feedforward information is prominent in the areas: in the case of negative offsets a possible increase in decoding accuracy indicates the importance of feedback information, i.e., the current activity of the neurons explains an event that has already happened; in the case of positive offsets the increase in accuracy is related to a greater prominence of the elaboration of the movement plan that will be realized in subsequent times. While evaluating differences across offsets for each coordinate, area and monkey, significant differences ($p < 0.01$, Friedman test) across offsets were only found for V6A for the x-coordinate for both monkeys, where a positive linear trend from negative to positive offsets was observed. By focusing only on the performance obtained with zero-lag, we observed that PE scored lower R2 than other areas ($p<0.01$, Wilcoxon signed-rank test corrected for multiple tests using the Benjamini-Hochberg procedure), for all monkeys and coordinates (see statistical analysis results displayed as solid bars in Figure 7.8b).

7.4. DISCUSSION

In this study we explored the possibility to predict reaching goals and movement trajectories by decoding the activity of single neurons recorded from different areas of the SPL. To decode neural signals, CNNs were used to extract temporal dynamics via temporal convolutions, and their design was defined by automatic hyper-parameter search using BO. This is particularly relevant as in the literature [24,26,27] DNN designs are commonly defined by empirical evaluation (i.e., test a limited bunch of configurations and using the best performing one) which is a time-consuming process, often leading to sub-optimal DNN configurations. CNNs were used in this study with two main purposes: i) leveraging temporal convolution to better characterize encoding dynamics in the different areas of SPL reaching network; ii) validating for the first time (to best knowledge of the authors) CNNs in decoding spiking activity, as CNNs among DNNs (e.g., respect to RNNs) achieve good performance and explainability, while keeping a lower number of trainable parameters. Finally, we probed CNNs superiority compared to simpler linear classifiers. Decoding results allowed to uncover the amount and characteristics of information each area contains about the external variables (end-point position and effector trajectory) and the differences across the different areas.

Signals collected from neurons of SPL were predictive of reaching targets also before the movement onset. Decoding performances were different between areas: while the decoding accuracies for V6A and PEc were very similar (over 80% of accuracy in detecting the correct spatial position), PE diverged by obtaining lower values (lower than 40%), suggesting a different role in the circuit for the different areas. Similarly, moving from decoding reaching goals to movement trajectories, V6A and PEc maintained good R2 values, PE regained some points in the reconstruction of the depth (y) and elevation (z) components of the movement while reconstruction of the direction (x) remained more difficult. Neural networks have proven to be robust and easily adaptable to the required task, preferring simple architectures (see the more frequent optimal configuration in Section 7.3.2) and therefore quickly trainable and generalizable, being able to capture the dynamics of neuronal activity.

7.4.1. A visual to somatosensory gradient over the network is reflected in the decoding accuracy.

The posterior parietal cortex sits in a crucial node for integration of sensory stimuli to guide action receiving visual input, somatosensory and proprioceptive feedback, and afferent motor copy from premotor cortex [51,52]. In agreement with this, electrophysiological studies that probed different areas of the PPC found different levels of activation for these neurons, mostly linked to visual stimuli for caudal regions close to extrastriate cortex, tactile and proprioceptive moving rostrally toward somatosensory cortex. The network we studied bridges visual and somatic domains, with area V6A bordering the extrastriate visual area V6 [9,53], PE bordering Brodmann's area 2 and PEc in between [20]. In agreement with this, the number of neurons modulated by the spatial goal position (see sliding ANOVAs of Figure 7.3) was already high in V6A in the first part of the trial (epoch 1), being the visual input of great relevance in V6A. The number of neurons modulated by goal position is maximal in PE in the second part of the trial, that was associated with the execution of the reaching movement (epoch 4-5). Although even in the second part of the task the percentage of modulated neuron of PE was not exceeding

those of V6A and PEc, the stronger representation of the goal position in the movement epoch reflects a stronger somatomotor than visuospatial representation in PE. The neural activity from the first visualization of the target to the movement onset has been associated with the elaboration of the motor plan, the maintenance of spatial attention toward relevant targets and the integration of sensorimotor inputs [52,54].

Moving from the previous ANOVA analysis, we then performed a more sensitive decoding analysis using CNN; the latter, taking into account neural dynamics and non-linearity, can guide a more robust comparison. Given the link between preparatory activity and the spatial positions of the targets, decoders successfully extracted the correct position given the preparatory spiking activity. Our results suggest that V6A and PEc strongly encode visuospatial information already in the first part of the task, enabling the decoders to easily extract useful information about the goal spatial positions. During the planning epoch, visuospatial inputs are converted to visuomotor signals which are required to guide execution phase. During this second part of the task, signals related to goal position remained strong and easily decodable. PE in comparison, presents a less pronounced activation in relation to the task tested, nevertheless during the execution phase stronger sensorimotor signals can be decoded by CNNs. It is worth noticing that gaining accuracy in the movement phases, in all areas and especially in PE, is a peculiarity of the CNN classifier, not exhibited by the linear NB classifier (see Figure 7.7); this may indicate that CNNs are more apt to catch the richness of information contained in these areas. Ultimately, PE and PEc both part of the Brodmann's area 5 [55] appear more different than expected: the activity of PEc is much more similar to V6A, part of Brodmann's area 7, and should therefore be considered part of the latter area 7. This idea, advanced by [51] is supported by present data.

7.4.2. Decoding movement goals and trajectories from PPC

Decoding of reaching goals is particularly efficient in different areas of the PPC especially from areas of the dorso medial network. The Parietal Reach Region (PRR) has been used as a source of these signals in several studies [56–59]. Interestingly, the PRR is close to areas V6A and PEc and mostly overlap with area MIP (anterior bank of the medial intraparietal sulcus) [38]. Homologue of PRR in humans, together with the more lateral anterior intraparietal area (AIP), were used to decode motor imagery in a center-out task by a tetraplegic implanted patient [60]. Thus, the interest in decoding movement intention from PPC remains high in light of the possibility to extract several parameters related to cognitive processing rather than simpler motor kinematics [61]. Most of the studies used a task with reaching movements towards a monitor placed in front of the subject, without studying the movement in depth. Conversely, in our study the movements were made on three different degrees of depth simulating more naturalistic movements. Several pieces of evidence support the diversity of networks processing direction and depth information, with different percentages of cells modulated in depth and direction for the different areas tested [13,15,62,63]. Depth encoding is plausibly stronger for areas that rely more on proprioceptive (such as PE) rather than visual [18]. We did not find different accuracies by decoding the two components separately when we decoded reaching goals (data not shown). While decoding the 9 reaching targets (supervised problem 1), not only movement attributes, planning and execution could be exploited in the

learning system, but also spatial attention, sensory feedback, and movement imagery. All these types of information are known to be encoded in PPC (Section 7.4.1) and contribute to the generation of patterns in the discharge of neurons, patterns that can be extracted from the neural network and mapped to the classes corresponding to spatial locations. We then tried to predict the trajectories of hand position (supervised problem 2) from the population activity of neurons providing the algorithm with past neural activity up to 300 ms before the current movement (corresponding to an offset=0, see Section *Supervised problem 2: Hand trajectory decoding* and Figure 7.8b). Unfortunately, the real trajectories were not available, so we have reconstructed the plausible trajectories of movement semi-synthetically (see Section *Supervised problem 2: Hand trajectory decoding*), from the pressure of the home button to the reaching movement up to the holding of goal position. Remarkably, while V6A and PEc R2s were high (over 0.6) obtaining a good trajectory reconstruction, R2s of PE were lower (see Figure 7.8b), especially for the x-coordinate (corresponding to the direction of movement) supporting the view of preferential depth (y) encoding from rostral SPL. Our task was ideal for testing visuospatial transformations and probably little activates areas more devoted to somatomotor control. Nevertheless, it is plausible that the semi-synthetic trajectories we used, forcing a non-natural straight-line trajectory, could be not optimal for decoding. While reaching goal location and trajectory decoding were good for both V6A and PEc, and no particular difference emerges between the two methods, trajectory decoding from PE seems to perform slightly better than classification (especially for depth and elevation). This could be related to the specialization of the area PE in dealing with proprioceptive signals, so that information of the absolute position of the target (used for classification problem) is scarcely useful, and signals are more related to the movement of the limbs than visuospatial representation.

7.4.3. Feedforward model

To perform rapid, targeted movements, our brain must rely on a feedforward predictive model given the latency of incoming sensory feedback signals. One of the theories that is gaining momentum is that the brain must continuously integrate the state of the environment and the body into a feedback control loop to perform congruent movements in real time [64]. The SPL fits nicely into this framework with caudal regions encoding the environment in relation to the body and more rostral regions encoding the state of the body [22]. Within this model it is possible to frame the generation and deployment of the trajectories of movement. We found that it is possible to predict the instantaneous position of hand in the space by providing the activity of neurons in a short prior interval, but also with increasing lags. Negative lags extract features related to sensory feedback (sensory outcomes of action), positive lags suggest the existence of a predictive model or motion planning. Mulliken and colleagues [23] observed how PRR neurons encode for either the movement angle or goal angle (or both), finding single-cell preferences for encoding future states (positive lag), past states (negative lag), and many cells that represented the current state (zero lag, in particular for a task that used an obstacle on the trajectory requiring a more dynamic control), demonstrating the existence of the feedforward model. Our decoding analysis loses sensitivity to a single neuron combing all contributions at the population level but still gives us clues about the existence along the entire network of the SPL of the running feedforward model. The fact that

the areas we studied simultaneously contain a representation of sensory feedback and signals related to the planning of future movements, supports the existence of an inner model that compares the expected outcome of an action with the real outcome, even if with intrinsic latency. While in most cases the decoding accuracy was not affected by the different offsets, only the x component for the V6A area shows significant dependence (Figure 7.8b). We think that this reflects two properties of the network we are studying. Previous work from our lab found that direction information (x-axis) is processed earlier than depth (y-axis, see Hadjidimitrakis et al. [13]) that relies more on somatosensory signals arriving later as sensory feedback from the moving arm. In addition, somatosensory afferents are greater for more rostral areas (PEc and PE, see Introduction). Accordingly, V6A decoding accuracy is more affected by offsets that rely less on feedback signals (see lower accuracy for negative offset in Figure 7.8b), where a high accuracy is maintained using movement preparatory signals (positive offsets). The offset effect is less evident in PEc, with a trend that is not statistically significant and in PE it is not present (although it could be masked by the lower overall accuracy).

7.4.4. Convolutional neural networks for neural decoding

The revival of neural networks supported by a large variety of applications in computer vision has led to the widespread use of neural networks in various fields that can now benefit from advanced pattern recognition techniques [65]. We have borrowed techniques from time series analysis and have shown how neural networks are well suited for the study of neural dynamics. Since our CNN-based algorithm is able to extract and leverage temporal features, the decoding performance significantly increases (Figure 7.7) compared to a classic algorithm based on Naïve Bayesian.

Conversely to other studies [24,26,29], in this study we adopted CNNs to decode neural signals while automatically searching for its best configuration using BO. Therefore, it is worth remarking that the main CNN structural hyper-parameters were automatically optimized within the search space exploiting an automatic search algorithm, rather than manually select them based on a trial-and-error procedure. From BO, a shallow CNN architecture (i.e., $N_b = 1, N_c = 1$) with one separable convolutional layer and one fully-connected layer resulted optimal for decoding the neural activity during reaching. The adoption of a shallow CNN has the advantage of lower training times and good generalization with limit-sized datasets as the one adopted in this study, achieving high decoding performance both in classifying the target reaching end-point, and in predicting semi-synthetic trajectories. Indeed, results suggest that with the adopted optimal shallow CNN few trials (72 in the training set, see Section *Behavioral task*) are enough to train the networks with the chunking procedure (augmenting the training set up to 3384 examples, see Table 7.1), achieving high performance. Temporal patterns of the single separable convolutional layer led to an optimal decoding when extracted within a window of 100 ms (i.e., $F = 5$ bins). Interestingly, despite being a shallow CNN, the optimal architecture learned the highest number of features among the admitted values of hyper-parameter space (i.e., $K = 32$ feature maps were learned, see Table 7.2). Thus, instead of selecting a deep convolutional neural network (e.g., 2 blocks with 4 convolutional layers per block, corresponding to the maximum depth in the defined search space) and a low number of filters

per layer (e.g., 4 filters), which is a common design principle in computer vision applications [66], BO selected a simple shallow CNN learning the highest possible number of filters in the single convolutional layer included. Therefore, by analyzing these structural hyper-parameters, results suggest that the information contained in the input neural chunks did not require extracting high-level and more abstract features (e.g., as resulting from a deep CNN in deeper layers) to perform an accurate decoding, but rather learning many low-level and less abstract features directly from the raw input chunks was more beneficial. This result may depend on the fact that some high-level features have already been extracted upstream, from the visual and somatosensory processing flows that precede the posterior parietal lobule, i.e., the inputs to our network are not directly taken from the external world, as usually done in deep neural networks, but have been significantly pre-processed by the primary brain areas. Future applications in neural decoding could benefit in designing shallow but wide CNNs rather than deep and narrow CNNs. Furthermore, regarding regularization hyper-parameters, constraints such as kernel max-norm constraint (with $c = 1$) and dropout (selecting the highest dropout probability, i.e., $p = 0.5$, see Table 7.2) proved their utility improving the generalization in the addressed decoding tasks. Interestingly, batch normalization did not result as useful as the previous regularization methods (BO selected less frequently this regularizer). Thus, in perspective, neural decoders could benefits in applying the specific combination between kernel max-norm constraint and dropout (with a high dropout probability, e.g., set to 0.5 as in this study) to perform regularization.

Despite the main objective of this study was to propose a CNN architecture for decoding neural activity and enabling the analysis of three different PPC areas, some methodological aspects may prospectively have significant implications for BCI. First, the proposed CNN structure resulted from an automatic algorithm (BO) and resulted optimal in terms of performance on a separate validation set, significantly outperforming a linear classifier. Furthermore, the design included separable convolution that are lighter and more efficient than standard convolutions, [41], providing a neural network that is less prone to overfit small datasets and that produces a fast inference. That is, the proposed CNN resulted in an accurate, light, and efficient non-linear decoder of neurons' spiking rate that, in prospective, may find some applicability in BCI systems.

7.4.5. Future directions

Although we focused on CNNs in our analysis because we think they should be better explored as method for neural decoding thanks to their simplicity and interpretability [30], many studies used different implementations based on RNNs (as mentioned in the introduction) obtaining good results especially in the decoding of trajectories for practical applications (despite RNNs remain in many cases black boxes). RNNs are being trained to simulate the frontoparietal network of grasping (dorso-lateral pathway rather than dorso-medial of reaching), with artificial units resembling the neural activity of real neurons recorded in areas of grasping circuits. In such model, virtual lesion to the artificial network produced outputs similar to lesion/inactivation studies on monkeys [67]. Given the interconnections between the reaching and grasping networks with neurons sensitive to grip types found in V6A [68] and neurons sensitive to reach locations in anterior intraparietal area [69] it is expected that a similar

RNN architecture can also be applied to the dorso-medial network. Future developments include the adoption of RNN to compare with CNNs in a benchmark (with many datasets and decoding algorithms) and evaluate which decoding approach represents the best compromise between performance, training time and model size (i.e., number of trainable parameters). Furthermore, techniques aimed to improve the interpretability of the CNN (e.g., occlusion techniques and saliency maps [70]), recently exploited to investigate neural signatures in the electroencephalogram while decoding brain states [71,72], can be of value also to explain network's decision when decoding neurons' spiking rates. In particular, these explanation techniques could help to characterize the impacts of individual input neurons or subpopulations of neurons (e.g., at different locations) inside each specific area in the decoding process, as well as the importance of specific time bins, contributing to understand their role at the level of brain network dynamics. Working on these two points, i.e., developing algorithms that decode efficiently and accurately neural dynamics, and explaining decoding decision could bring great benefits. First, it may further increase our knowledge about the link between neural activity and behavioral outcome; second, and prospectively, it may contribute to advance BCI technologies by driving improvements aimed to maximize brain information extraction and better brain-computer communication.

Finally, it is important to stress that neural decoders as the ones proposed here are of significance to determine the amount and nature of information neural populations contain about specific external variables, but are not designed for mechanistic interpretation, i.e., for explaining the neural mechanisms underlying multisensory and sensorymotor integration in PPC for guide actions. To this aim, biologically inspired neural networks are needed, designed to functionally and structurally resemble specific parts of the brain and to implement more biological learning rules than back-propagation. Data-driven deep learning approaches and biologically constrained interpretative networks are complementary approaches that can both boost a better comprehension of how information is encoded and processed in the brain and each one can support the advancement of the other; for example, a better description of the information encoded by the different neural populations gained by CNNs decoders may guide the design of interpretative models [73].

7.5. CONCLUSIONS

We decoded the activity of neurons from three areas of the reaching network within the superior parietal lobule of macaque, V6A, PEc, and PE to reconstruct the position of the goal in space and the trajectory required to accomplish the reaching. CNNs were used as neural decoders and proved to accurately decode both the reaching target and 3D hand position. The optimal design of the CNNs, as obtained with hyper-parameter search, resulted in shallow (but wide) architectures with only one hidden separable convolutional layer. While the more caudal V6A and PEc encoded more strongly the position of the target in space (decoding accuracy was already good at the presentation of the target) the area PE, more rostral, was weaker in this representation, its accuracy reaching a peak during the execution of the movement. This supports a model of the PPC where the more caudal areas represent the body-environment relationship and the more rostral areas the effects of the action on the body. The results can be framed in the role played by PPC in the neural control of reaching movements. New generations of BCIs can gain benefits from a better combined study between system neuroscience and renewed deep learning technologies.

7.6. REFERENCES

[1] Andersen R A and Cui H 2009 Intention, Action Planning, and Decision Making in Parietal-Frontal Circuits *Neuron* **63** 568–83

[2] Medendorp W P and Heed T 2019 State estimation in posterior parietal cortex: Distinct poles of environmental and bodily states *Progress in Neurobiology* **183** 101691

[3] Ungerleider L G and Mishkin M 1982 Two cortical visual systems *Analysis of Visual Behavior* 549–86

[4] Goodale M A and Milner A D 1992 Separate visual pathways for perception and action. *Trends in neurosciences* **15** 20–5

[5] Galletti C, Kutz D F, Gamberini M, Breveglieri R and Fattori P 2003 Role of the medial parieto-occipital cortex in the control of reaching and grasping movements *Experimental brain research* vol 153 pp 158–70

[6] Rizzolatti G and Matelli M 2003 Two different streams form the dorsal visual system: Anatomy and functions *Experimental Brain Research* vol 153 pp 146–57

[7] Pisella L, Sergio L, Blangero A, Torchin H, Vighetto A and Rossetti Y 2010 Erratum to Optic ataxia and the function of the dorsal stream: Contributions to perception and action, [Neuropsychologia, (2009), 47, 14, 3033-3044] *Neuropsychologia*

[8] Karnath H-O and Perenin M-T 2005 Cortical control of visually guided reaching: evidence from patients with optic ataxia. *Cerebral cortex (New York, N.Y. : 1991)* **15** 1561–9

[9] Galletti C, Battaglini P P and Fattori P 1995 Eye Position Influence on the Parieto-occipital Area PO (V6) of the Macaque Monkey *European Journal of Neuroscience* **7** 2486–501

[10] Hadjidimitrakis K, Breveglieri R, Placenti G, Bosco A, Sabatini S P and Fattori P 2011 Fix your eyes in the space you could reach: neurons in the macaque medial parietal cortex prefer gaze positions in peripersonal space. ed P L Gribble *PloS one* **6** e23335

[11] Hadjidimitrakis K, Breveglieri R, Bosco A and Fattori P 2012 Three-dimensional eye position signals shape both peripersonal space and arm movement activity in the medial posterior parietal cortex *Frontiers in Integrative Neuroscience* **6** 37

[12] Fattori P, Kutz D F, Breveglieri R, Marzocchi N and Galletti C 2005 Spatial tuning of reaching activity in the medial parieto-occipital cortex (area V6A) of macaque monkey *European Journal of Neuroscience* **22** 956–72

[13] Hadjidimitrakis K, Bertozzi F, Breveglieri R, Bosco A, Galletti C and Fattori P 2014 Common neural substrate for processing depth and direction signals for reaching in the monkey medial posterior parietal cortex *Cereb Cortex* **24** 1645–57

[14] Galletti C, Breveglieri R, Lappe M, Bosco A, Ciavarro M and Fattori P 2010 Covert shift of attention modulates the ongoing neural activity in a reaching area of the macaque dorsomedial visual stream. *PloS one* **5** e15078

[15] Hadjidimitrakis K, Dal Bo' G, Breveglieri R, Galletti C and Fattori P 2015 Overlapping representations for reach depth and direction in caudal superior parietal lobule of macaques. *Journal of neurophysiology* **114** 2340–52

[16] Gamberini M, Dal Bò G, Breveglieri R, Briganti S, Passarelli L, Fattori P and Galletti C 2018 Sensory properties of the caudal aspect of the macaque's superior parietal lobule *Brain Structure and Function* **223** 1863–79

[17] Seelke A M H, Padberg J J, Disbrow E, Purnell S M, Recanzone G and Krubitzer L 2012 Topographic maps within brodmann's area 5 of macaque monkeys *Cerebral Cortex*

[18] De Vitis M, Breveglieri R, Hadjidimitrakis K, Vanduffel W, Galletti C and Fattori P 2019 The neglected medial part of macaque area PE: segregated processing of reach depth and direction *Brain Structure and Function* **224** 2537–57

[19] Cui H and Andersen R A 2011 Different Representations of Potential and Selected Motor Plans by Distinct Parietal Areas *Journal of Neuroscience* **31** 18130–6

[20] Breveglieri R, Galletti C, Monaco S and Fattori P 2008 Visual, Somatosensory, and Bimodal Activities in the Macaque Parietal Area PEc *Cerebral Cortex* **18** 806–16
[21] Land M F 2014 Do we have an internal model of the outside world? *Philosophical Transactions of the Royal Society B: Biological Sciences* **369** 20130045
[22] Medendorp W P and Heed T 2019 State estimation in posterior parietal cortex: Distinct poles of environmental and bodily states *Progress in Neurobiology* **183** 101691
[23] Mulliken G H, Musallam S and Andersen R A 2008 Forward estimation of movement state in posterior parietal cortex *Proceedings of the National Academy of Sciences* **105** 8170–7
[24] Glaser J I, Benjamin A S, Chowdhury R H, Perich M G, Miller L E and Kording K P 2020 Machine Learning for Neural Decoding *eneuro* **7** ENEURO.0506-19.2020
[25] Yann LeCun, Yoshua Bengio G H 2015 Deep learning (2015), Y. LeCun, Y. Bengio and G. Hinton *Nature*
[26] Sussillo D, Nuyujukian P, Fan J M, Kao J C, Stavisky S D, Ryu S and Shenoy K 2012 A recurrent neural network for closed-loop intracortical brain–machine interface decoders *Journal of Neural Engineering* **9** 026027
[27] Tseng P-H, Urpi N A, Lebedev M and Nicolelis M 2019 Decoding Movements from Cortical Ensemble Activity Using a Long Short-Term Memory Recurrent Network *Neural Computation* **31** 1085–113
[28] Sanchez J C, Erdogmus D, Nicolelis M A L, Wessberg J and Principe J C 2005 Interpreting spatial and temporal neural activity through a recurrent neural network brain-machine interface *IEEE Transactions on Neural Systems and Rehabilitation Engineering*
[29] Shah S, Haghi B, Kellis S, Bashford L, Kramer D, Lee B, Liu C, Andersen R and Emami A 2019 Decoding Kinematics from Human Parietal Cortex using Neural Networks *International IEEE/EMBS Conference on Neural Engineering, NER*
[30] Tjoa E and Guan C 2020 A Survey on Explainable Artificial Intelligence (XAI): Toward Medical XAI *IEEE Transactions on Neural Networks and Learning Systems* 1–21
[31] Yamins D L K, Hong H, Cadieu C F, Solomon E A, Seibert D and DiCarlo J J 2014 Performance-optimized hierarchical models predict neural responses in higher visual cortex *Proceedings of the National Academy of Sciences* **111** 8619–24
[32] Craik A, He Y and Contreras-Vidal J L 2019 Deep learning for electroencephalogram (EEG) classification tasks: a review *Journal of Neural Engineering* **16** 031001
[33] Simões M, Borra D, Santamaría-Vázquez E, Bittencourt-Villalpando M, Krzemiński D, Miladinović A, Schmid T, Zhao H, Amaral C, Direito B, Henriques J, Carvalho P and Castelo-Branco M 2020 BCIAUT-P300: A Multi-Session and Multi-Subject Benchmark Dataset on Autism for P300-Based Brain-Computer-Interfaces *Frontiers in Neuroscience* **14**
[34] Breveglieri R, Hadjidimitrakis K, Bosco A, Sabatini S P, Galletti C and Fattori P 2012 Eye position encoding in three-dimensional space: integration of version and vergence signals in the medial posterior parietal cortex. *The Journal of neuroscience : the official journal of the Society for Neuroscience* **32** 159–69
[35] Galletti C, Fattori P, Kutz D F and Gamberini M 1999 Brain location and visual topography of cortical area V6A in the macaque monkey. *European Journal of Neuroscience* **11** 575–82
[36] Galletti C, Gamberini M, Kutz D F, Baldinotti I and Fattori P 2005 The relationship between V6 and PO in macaque extrastriate cortex *European Journal of Neuroscience*
[37] Luppino G, Ben Hamed S, Gamberini M, Matelli M and Galletti C 2005 Occipital (V6) and parietal (V6A) areas in the anterior wall of the parieto-occipital sulcus of the macaque: A cytoarchitectonic study *European Journal of Neuroscience* **21** 3056–76

[38] Snyder L H, Batista a P and Andersen R a 1997 Coding of intention in the posterior parietal cortex. *Nature* **386** 167–70
[39] Kutz D F, Marzocchi N, Fattori P, Cavalcanti S and Galletti C 2005 Real-Time Supervisor System Based on Trinary Logic to Control Experiments With Behaving Animals and Humans *Journal of Neurophysiology* **93** 3674–86
[40] Kutz D F, Fattori P, Gamberini M, Breveglieri R and Galletti C 2003 Early- and late-responding cells to saccadic eye movements in the cortical area V6A of macaque monkey *Experimental brain research* **149** 83–95
[41] Chollet F 2017 Xception: Deep learning with depthwise separable convolutions *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*
[42] Ioffe S and Szegedy C 2015 Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift *32nd International Conference on Machine Learning, ICML 2015*
[43] Srivastava N, Hinton G, Krizhevsky A, Sutskever I and Salakhutdinov R 2014 Dropout: A simple way to prevent neural networks from overfitting *Journal of Machine Learning Research*
[44] Filippini M, Breveglieri R, Hadjidimitrakis K, Bosco A and Fattori P 2018 Prediction of Reach Goals in Depth and Direction from the Parietal Cortex *Cell Reports* **23** 725–32
[45] Roy A C, Paulignan Y, Farnè A, Jouffrais C and Boussaoud D 2000 Hand kinematics during reaching and grasping in the macaque monkey *Behavioural Brain Research* **117** 75–82
[46] Snoek J, Larochelle H and Adams R P 2012 Practical Bayesian Optimization of Machine Learning Algorithms *Advances in Neural Information Processing Systems*
[47] Kingma D P and Ba J 2014 Adam: A Method for Stochastic Optimization *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*
[48] Bergstra J, Bardenet R, Bengio Y and Kégl B 2011 Algorithms for hyper-parameter optimization *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011, NIPS 2011*
[49] Clevert D A, Unterthiner T and Hochreiter S 2016 Fast and accurate deep network learning by exponential linear units (ELUs) *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*
[50] Benjamini Y and Hochberg Y 1995 Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing *Journal of the Royal Statistical Society: Series B (Methodological)* **57** 289–300
[51] Gamberini M, Passarelli L, Fattori P and Galletti C 2020 Structural connectivity and functional properties of the macaque superior parietal lobule *Brain Structure and Function* **225** 1349–67
[52] Andersen R A and Cui H 2009 Intention, action planning, and decision making in parietal-frontal circuits. *Neuron* **63** 568–83
[53] Matelli M, Govoni P, Galletti C, Kutz D F and Luppino G 1998 Superior area 6 afferents from the superior parietal lobule in the macaque monkey. *The Journal of comparative neurology* **402** 327–52
[54] Galletti C and Fattori P 2018 The dorsal visual stream revisited: Stable circuits or dynamic pathways? *Cortex* **98** 203–17
[55] Pandya D N and Seltzer B 1982 Intrinsic connections and architectonics of posterior parietal cortex in the rhesus monkey. *The Journal of comparative neurology* **204** 196–210
[56] Musallam S, Corneil B D, Greger B, Scherberger H and Andersen R 2004 Cognitive control signals for neural prosthetics. *Science (New York, N.Y.)* **305** 258–62

[57] Andersen R A, Hwang E J and Mulliken G H 2010 Cognitive neural prosthetics. *Annual review of psychology* **61** 169–90, C1-3
[58] Hauschild M, Mulliken G H, Fineman I, Loeb G E and Andersen R A 2012 Cognitive signals for brain-machine interfaces in posterior parietal cortex include continuous 3D trajectory commands. *Proceedings of the National Academy of Sciences of the United States of America* **109** 17075–80
[59] Mulliken G H, Musallam S and Andersen R A 2008 Decoding trajectories from posterior parietal cortex ensembles. *The Journal of neuroscience : the official journal of the Society for Neuroscience* **28** 12913–26
[60] Aflalo T, Kellis S, Klaes C, Lee B, Shi Y, Pejsa K, Shanfield K, Hayes-Jackson S, Aisen M, Heck C, Liu C and Andersen R A 2015 Neurophysiology. Decoding motor imagery from the posterior parietal cortex of a tetraplegic human. *Science (New York, N.Y.)* **348** 906–10
[61] Andersen R A, Kellis S, Klaes C and Aflalo T 2014 Toward more versatile and intuitive cortical brain-machine interfaces. *Current biology : CB* **24** R885-97
[62] Crawford J D, Henriques D Y P and Medendorp W P 2011 Three-Dimensional Transformations for Goal-Directed Action *Annual Review of Neuroscience* **34** 309–31
[63] Tramper J J and Gielen C C A M 2011 Visuomotor Coordination Is Different for Different Directions in Three-Dimensional Space *Journal of Neuroscience* **31** 7857–66
[64] Todorov E 2004 Optimality principles in sensorimotor control *Nature Neuroscience* **7** 907–15
[65] Richards B A, Lillicrap T P, Beaudoin P, Bengio Y, Bogacz R, Christensen A, Clopath C, Costa R P, de Berker A, Ganguli S, Gillon C J, Hafner D, Kepecs A, Kriegeskorte N, Latham P, Lindsay G W, Miller K D, Naud R, Pack C C, Poirazi P, Roelfsema P, Sacramento J, Saxe A, Scellier B, Schapiro A C, Senn W, Wayne G, Yamins D, Zenke F, Zylberberg J, Therien D and Kording K P 2019 A deep learning framework for neuroscience *Nature Neuroscience* **22** 1761–70
[66] Simonyan K and Zisserman A 2015 Very deep convolutional networks for large-scale image recognition *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*
[67] Michaels J A, Schaffelhofer S, Agudelo-Toro A and Scherberger H 2020 A goal-driven modular neural network predicts parietofrontal neural dynamics during grasping *Proceedings of the National Academy of Sciences* **117** 32124–35
[68] Fattori P, Raos V, Breveglieri R, Bosco A, Marzocchi N and Galletti C 2010 The dorsomedial pathway is not just for reaching: grasping neurons in the medial parieto-occipital cortex of the macaque monkey. *The Journal of neuroscience : the official journal of the Society for Neuroscience* **30** 342–9
[69] Lehmann S J and Scherberger H 2013 Reach and gaze representations in macaque parietal and premotor grasp areas. *The Journal of neuroscience : the official journal of the Society for Neuroscience* **33** 7038–49
[70] Simonyan K, Vedaldi A and Zisserman A 2014 Deep inside convolutional networks: Visualising image classification models and saliency maps *2nd International Conference on Learning Representations, ICLR 2014 - Workshop Track Proceedings*
[71] Borra D, Fantozzi S and Magosso E 2020 Interpretable and lightweight convolutional neural network for EEG decoding: Application to movement execution and imagination *Neural Networks* **129** 55–74
[72] Borra D, Fantozzi S and Magosso E 2021 A Lightweight Multi-Scale Convolutional Neural Network for P300 Decoding: Analysis of Training Strategies and Uncovering of Network Decision *Frontiers in Human Neuroscience* **15** 304

[73] Kay K N 2018 Principles for models of neural information processing *NeuroImage* **180** 101–9

CONCLUSIONS

The investigations carried out in this PhD work aimed to overcome some limitations currently affecting CNN-based decoding of neural time series; these limitations mainly regard the network architecture, training strategy, feature interpretation, and the decoding of neurons' spiking rate.

Concerning the network architecture, the results of the present research have shown that the optimal CNN structure, in terms of performance, exploited less features to learn in the temporal and spatial domains in case of within-subject decoders (i.e., decoders trained with subject-specific examples), both for P300 and motor decoding from the EEG. Conversely, when the set of the training examples included more variability, e.g., when designing cross-subject decoders, the optimal structure did benefit from learning more features in the temporal and spatial domains. In addition, when designing deeper temporal convolutions, benefits, in terms of performance, resulted from including a multi-scale temporal feature learning, i.e., learning temporal filters having different size, and, thus, learning temporal features at multiple time scales over the input. These analyses and the obtained results are valuable to better clarify how to design the network layers, including deeper temporal convolution layers, by suggesting the proper setting of their hyper-parameters. Lastly, both for P300 and motor decoding from the EEG, the obtained results suggested that the proposed light architectures outperformed heavier and deeper neural networks presented in the literature, proving that a careful design of CNNs should be performed to achieve higher performance on unseen data when handling small datasets of neural time series, as commonly available in practice.

The investigation of multiple training strategies has allowed to obtain an extensive validation of CNNs as accurate EEG decoders, both in P300 and motor decoding problems. Furthermore, transfer learning, i.e., transferring the knowledge on a new subject from a model previously trained on other subjects, significantly increased the performance, especially in the very-low training data regime (i.e., in case of only a few or no training trials). This enables the design of accurate decoders (when based on CNNs) also in conditions where extremely compact training sets are available (calibration-limited condition), or even in absence of any training example (calibration-free condition).

In addition, the issue of interpreting the features learned by CNNs was extensively investigated in this research, with reference to P300 and motor decoding from the EEG. This analysis was performed by using both an input explanation technique and intermediate explanation technique, with the aim of increasing our knowledge about the neural features associated to P300 and of motor correlates in the spatio-temporal and frequency domains. It is worth noticing that, to enable intermediate explanations, we developed interpretable CNNs (incorporating interpretability directly inside the structure). These CNNs, besides being lighter and more interpretable, performed on par or even outperformed the non-interpretable solutions of the state-of-the-art across the presented chapters. This is of great relevance, as the results of this research also point out that an accurate DL-based decoder can be designed interpretable without sacrificing its decoding capabilities. Input explanations were performed to study the P300 response and proved to better highlight and better enhance relevant P300 features in the spatio-temporal domain even at the single-trial or single-subject level, overcoming the limitations of a traditional analysis (ERP analysis) that requires averaging across multiple trials

and subjects. Advantages compared to a traditional analysis were also derived from intermediate explanations, that proved to better highlight P300 features related to autistic subjects in the frequency and spatial domains. Furthermore, intermediate explanations provided also useful representations related to motor correlates. Overall, the CNN-based EEG decoders, not only serve as accurate classifiers of different brain and behavioral states, but, if appropriately designed and inspected in their learned features, may represent valuable analysis tools able to highlight and disclose meaningful features of the underlying neural processes.

Lastly, CNNs were introduced to decode neurons' spiking rate from single-cell recordings of monkeys. Light CNNs outperformed the current ML state-of-the art and represented a valid alternative to decoders based on FCNNs and RNNs that are heavier and require longer training times.

In perspective, the present research may be useful not only for practical engineering applications, by adopting CNNs to increase the performance of BCI decoders and to reduce BCI calibration times (e.g., via transfer learning), but also for theoretical neuroscience knowledge. In this regard, by taking advantage of the learned knowledge of CNNs, it is possible to analyze in a data-driven way how the brain or behavioral states under investigation are encoded in the neural activity, by highlighting their more meaningful features in the temporal, spatial and frequency domains without relying on a priori assumptions about the underlying neural processes. These features may be used, in perspective, to automatically design novel EEG biomarkers for neurological or neurodevelopmental disorders.

As described above, the present research brings some innovations and improvements. However, it also suffers from some limitations that can be addressed in the near future. First, the decoding problems were addressed, across chapters, adopting different decoders often specialized to solve a specific decoding problem (e.g., P300 decoding). In particular, we can observe that the CNN structures adopted in the different chapters shared a common sub-network consisting of the first temporal convolutional layer immediately followed by a spatial convolutional layer, but they also presented relevant differences. Indeed, the first temporal convolution was designed interpretable or not depending on the cases. Furthermore, the adopted structures differed in the design of subsequent layers, as they either did or did not include another convolutional layer and the latter, when included, learned temporal features at a single or multiple time scales depending on the cases. It could be of interest to investigate a unique CNN structure able to perform well across multiple recording paradigms and datasets, to provide a unique decoder and CNN-based analysis tool that could be applied to decode and analyze many behavioral and brain states. Second, CNN-based decoders and analyses were applied in this research on datasets consisting of a limited number of subjects (≤ 25). This could have limited the validation of CNNs for EEG decoding and analysis, not only on healthy subjects but especially on patients where the disorder involves a high heterogeneity across subjects, e.g., the high heterogeneity of the behavioral symptoms in autism (see Chapter 3). Third, the analyses of the features learned by the CNNs while decoding neural time series was performed only on EEG signals and not on single-neuron recordings.

Therefore, future investigations will focus on: i) the design of a unique CNN architecture that generalizes well across paradigms and tasks; ii) the adoption of larger datasets to provide a more robust validation of CNNs as neural decoders and data-driven analysis tools, able to

highlight robust neural signatures underlying behavioral and perceptual phenomena; iii) the study of the features learned by the CNN from single-cell recordings, to validate the data-driven analysis approach presented in this Thesis when using a different recording modality of the neural activity. In addition to these future developments related to the main limitations of the present research, the analyses described in this Thesis will be extended as follows. As described in the Introduction, CNN-based decoders generally achieved high decoding performance without handling artifacts (e.g., as presented in Chapter 1), leaving the learning system free to explore raw or minimally pre-processed EEG signals. However, it remains unclear to which extent CNNs and different CNN structures are robust to artifacts. Therefore, the analysis on the proper design of CNNs will be extended by also evaluating CNN robustness to artifacts, by analyzing CNN decoding capabilities and CNN designs while progressively cleaning EEG signals from artifacts. Lastly, the CNN-based analyses of the EEG at the scalp level presented in this Thesis will be extended to study neural features of source activations, as computed from the EEG, within region of interests.