Deep Pinsker and James-Stein Neural Networks for Brain-Computer Interfacing

Marko Angjelichinoski*, Mohammadreza Soltani*, John Choi†,
Bijan Pesaran†, and Vahid Tarokh*
*Department of Electrical and Computer Engineering, Duke University
†Center for Neural Science, New York University

Abstract

Objective. The theory of non-parametric regression has been shown to be useful in extracting relevant features from Local Field Potential (LFP) signals for decoding motor intentions. Yet, conventional designs of brain-computer interfaces predominantly rely on simple linear classification methods, circumventing deep neural networks due to training data scarcity. This paper leverages the robustness of several fundamental results in non-parametric regression - namely, Pinsker's theorem and its adaptively optimal counterpart due to James-Stein - to harness the potentials of deep learning in limited data setups. *Approach.* We propose a novel deep learning architecture that combines non-parametric regression for feature extraction, using Pinsker's and the James-Stein estimators, with a deep neural network for classification. *Main result and Significance.* We apply our deep Pinsker and James-Stein neural networks to the problem of decoding eye movement intentions from LFPs collected in macaque cortex at various cortical depths, while the animals perform memory-guided visual saccades to one of eight target locations. The results demonstrate that our deep learning architecture with non-parametric feature extraction is capable of achieving decoding accuracy of up to 96.5% at superficial cortical depths and up to 72% at deep cortical sites, making substantial improvement over state-of-the-art methods.

I. Introduction

A key challenge in neural engineering is the restoration of lost and/or impaired motor skills in subjects with chronic disabilities by means of neural prostheses [1]–[5]. At the heart of this challenge is designing reliable and robust *brain-computer interfaces (BCIs)* that utilize either non-invasive signal acquisition technology such as electroencephalography (EEG), or chronically implanted microelectrode arrays to collect signals directly from brain tissue [6], [7].

To foster successful practical deployment of neural prostheses, several important issues should be addressed. First, BCIs should generalize well in decoding motor intentions from neuronal activity signals recorded at different cortical sites/depths. Second, BCIs should generalize well across population of subjects that perform the same/similar tasks. Last but not least, BCIs should generalize well when trained with limited and even scarce training data; this is particularly important in invasive BCI configurations where acquiring training data is expensive and time-consuming endeavour [7], [8].

Conventional approaches for designing BCIs utilize the firing rates/patterns of individual neurons, as information-carrying features [2], [8], [9]. However, the ability to detect the multiunit spiking activity diminishes over time due to the sensor degradation. Recent advances have shown that local field potential (LFP) signals obtained by low-pass filtering the same wide-band neuronal activity signal from which the spikes are high-pass filtered, present a viable alternative for designing reliable BCIs with long-term outlook [8], [10]–[13]. In this regard, the theory of non-parametric regression has proven to be vital for the success of LFP-based neural decoders [14], [15]. This has led to the development of complex-valued spectrum-based feature extraction technique based on the famous Pinsker's theorem [16], [17]. As opposed to conventional feature extractors based on the spatial covariance matrix [18]–[22] or the power spectrum density (PSD) [12] of the LFP signals, Pinsker's feature extractor naturally incorporates both the amplitude *and* phase information inherently present in the LFP signal, leading to significant improvements in the accuracy of decoding motor intentions from LFP data [14], [15]. However, in these early studies, Pinsker's features are used along simple classification approaches such as the Linear Discriminant Analysis (LDA) [23], [24]. While the LDA leads, to some extent, to a robust BCI architecture, and it performs well even for scarce data sets, its performance is rather limited; that is, the decoding accuracy exhibits substantial variability across cortical depths with a trend to be noticeably poorer at deeper cortical sites [12], [14], [15].

The re-emergence of deep neural networks (DNNs) makes them an attractive choice for BCIs. Several recent works have reported some degree of success in this regard. The work presented in [25] applies deep convolutional neural network (CNN) for classification of motor imagery tasks using multichannel EEG data where the signals are first translated into time-frequency images using the short-time Fourier transform; these images are then fed as inputs to the CNN. In similar fashion, the authors in [26] apply CNN over the spatial covariance matrices of multichannel EEG data. An early work in [27] applies recurrent neural network (RNN) for predicting macaque arm

reaches from the spiking activity of large network of neurons recorded using invasive data acquisition methods. RNNs have been also used in [28] to capture the temporal variability of firing patters of the neurons recorded across long period of time, fostering successful cross-session decoding. Further progress has been reported in [29] where variational autoencoder (VAE) was used to model and predict single-trial firing patterns from large population of neurons. In addition to the aforementioned notable works, [30]–[32] provide comprehensive reviews of common classification methods in BCIs including DNNs.

DNNs are known to require extensive training data [33], [34] and it should be noted that in all of the works cited above, the available data is consistently large, which is the exception rather than the rule [7], [30]–[32]. Specifically, data sets in invasive BCI configurations are often quite limited due to two main reasons: (i) collecting sufficient amounts of training data via invasive electrophysiology techniques is time-consuming and expensive process [2], [7], [8], and (ii) neuronal activity signals are known to be highly non-stationary, which implies that their structural and statistical properties vary substantially over time, space and subjects [35], [36]; this limits the applicability of the training data collected over longer periods of time, in the sense that a BCI algorithm trained using data collected over given period (e.g. one day) is likely to be inapplicable to test data acquired later (e.g. the next day) [28]. As a result, BCIs are expected to be trained and re-trained over small batches of up-to-date data. These are important reasons why DNNs have been largely avoided in LFP-based BCIs.

Motivated by the robustness of the non-parametric feature extraction method based on Pinsker's theorem proposed in [14], [15] and the success of deep learning [33], [34], we propose a novel decoding architecture for BCIs, designed to combine the benefits of both methodologies and foster successful application of deep learning in limited data setups. The proposed method leverages the strength of the fundamental results from the theory of non-parametric regression, i.e., Pinsker's theorem as well as the blockwise extension of the James-Stein estimator [16], [17], and their proven ability to extract relevant and meaningful features from neural signals, and enable the DNN to be trained with limited, even scarce data sets. In particular, we consider two architectures. The first one uses the minimax-optimal function estimator due to Pinsker for feature extraction, followed by fully connected multilayer perceptron (MLP); we refer to this architecture as *deep Pinsker neural network (DPNN)*. The other architecture applies the James-Stein estimator, which is an adaptively minimax-optimal variant of Pinsker's estimator, valid over large sets of classes of smooth functions, followed by fully connected MLP and referred

to as *deep James-Stein neural network (DJSNN)*. We apply these architectures for decoding intended eye movement directions from LFP data collected from the prefrontal cortex (PFC) of two macaque monkeys performing memory-guided visual saccades to one of eight target locations on a screen. As demonstrated by the results of our investigations, a relatively shallow DNN when trained over Pinsker's features is powerful enough to significantly outperform simple decoding approaches based on linear methods and achieve peak decoding accuracy of up to 96.5% at superficial depths near the surface of the prefrontal cortex. Even more remarkably, the decoding accuracy of the DPNN at deep cortical sites peaks at 72%; these results are a substantial improvement over methods based on LDA classifiers.

II. Methods

We organize this section into three parts. In Section II-A we describe the experiment and the acquired data. Then, in Section II-B we review the theory of non-parametric regression including Pinsker's theorem and the blockwise formulation of the James-Stein estimator. Section II-C focuses on our main contribution and presents the deep learning architecture with non-parametric feature extraction.

A. Experimental Setup

We begin with a brief overview of the visual saccades experiment in Section II-A1. Section II-A2 discusses the data acquisition techniques. Details about the acquired data are given in Section II-A3. All experimental procedures were approved by the NYU University Animal Welfare Committee (UAWC).

1) Memory-guided visual saccades: Two adult male macaque monkeys (*M. mulatta*), referred to herein as Monkey A and Monkey S, were trained to perform a memory-guided visual saccade task. The setup consists of an LCD monitor on which targets are presented to the animal, see Fig. 1. The monkey initiates a trial by fixating a central target. Once the monkey maintains fixation for a baseline period, one of eight peripheral targets (drawn from the corners and edge midpoints of a square centered on the fixation target) is flashed on the screen for 300 milliseconds. For a random-duration memory period (1 - 1.5 seconds), the animal must maintain fixation on the central target until it is turned off. This event gives the monkey a visual cue to saccade to the remembered location of the peripheral target. If the monkey holds the gaze within a square window of the true target location, the trial is completed successfully. We only LFP

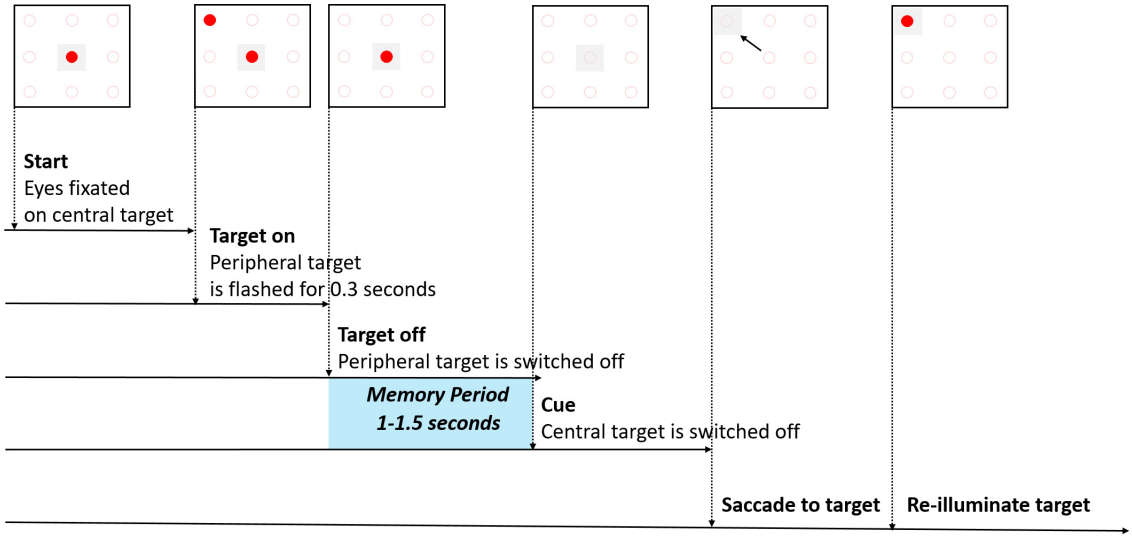


Fig. 1. Memory-guided visual saccade experiment. See Section II-A1 for details.

segments during the memory periods of a successful trials in our analyses since this epoch reflects information storage and generation of the resulting motor response.

2) Data acquisition: The animals were instrumented with a head restraint prosthesis that enables head position fixation and eye movement tracking. The recording chambers were surgically implanted in a craniotomy made over the right prearcuate gyrus of lateral PFC using MRI-guided stereotaxic techniques (Brainsight) in both animals. An electrode array consisting of $N = 32$ individually movable electrodes (20 mm travel, Gray Matter Research) was semi-chronically implanted into the recording chambers. Signals were initially recorded at 30 kHz and were subsequently downsampled to $\nu_S = 1$ kHz to obtain the LFPs. The initial position of each electrode (also referred to as channel, interchangeably) at the beginning of the experiment was receded 1 millimeter within the drive; after penetrating the dura and the pia, action potentials were first recorded at a mean depth (from the initial position) of 2.23 and 3.04 millimetres for Monkey A and Monkey S, respectively. As the experiment progressed, the positions of individual electrodes were gradually advanced deeper; the mean depth increase step was 34 and 100 microns for Monkey A and Monkey S, respectively. The reader interested in additional technical details regarding the materials and methods used to trace the action potential activity is referred to [12].

3) Data description: For a fixed electrode depth configuration, multiple trials were performed. Henceforth, a fixed configuration of electrode positions over which trials are collected is referred to as *electrode depth configuration (EDC)*. Each EDC is uniquely described by a 32-dimensional

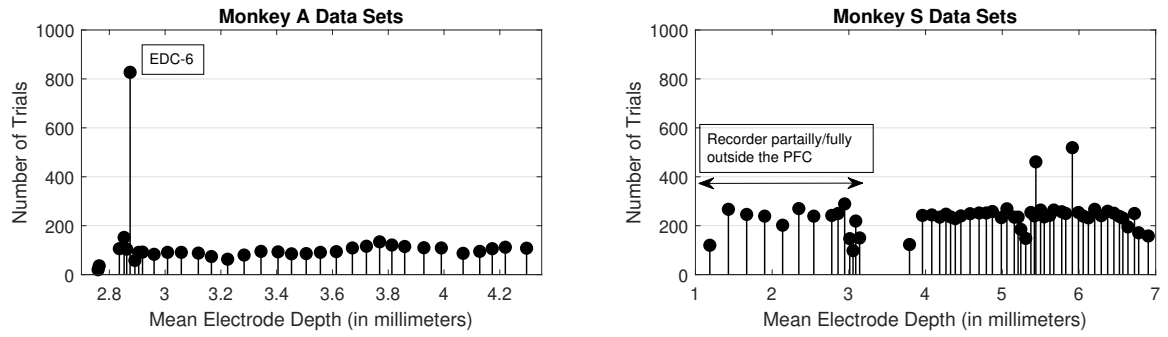


Fig. 2. The distribution of successful trials across EDCs. Trials were performed for 34 and 54 unique EDCs in Monkey A and Monkey S, respectively. The horizontal axis denotes the mean electrode depth computed by averaging all entries in the recorded EDC depth vectors. The initial 14 EDCs in Monkey S are fully/partially outside the PFC.

real vector; each entry in the vector contains the depth position of an individual electrode (in millimeters) with respect to its initial position (and not with respect to the depth at which action potentials were first detected). The subjects were trained across several days and recording sessions. As also described in Section II-A2, over the course of the experiment, the positions of the electrodes were gradually advanced into the PFC. The experiment was performed over a total of 34 and 55 EDCs for Monkey A and Monkey S, respectively, during which multiple trials were collected. The distribution of successful trials across EDCs is visually depicted in Fig. 2 where the horizontal axis denotes the mean electrode depth, computed as a simple average of the entries of the EDC vector; the indexing of the EDCs used here reflects the temporal progression of the experiment; namely, the trials for EDC-i were collected earlier in time than the trials for EDC-j if $j > i$. The total number of trials across all EDCs is 3922 for Monkey A and 13064 for Monkey S. The respective averages are ≈ 90 and ≈ 250 trials per EDC for Monkey A and Monkey S, respectively (with the exception of EDC-6 in Monkey A for which a total of 827 trials were collected across 10 recording sessions). It should be noted that the experiment for Monkey S began before action potential were detected, i.e., before the electrodes penetrated the surface of the PFC, and the recordings for the first 14 EDCs were taken while some (or all) of the electrodes were still outside the PFC [12].

B. Non-parametric Feature Extraction Methods: Theoretical Foundations

We present a brief overview of the theory of non-parametric regression that underlines our feature extraction approaches; for more detailed technical discussions, the interested reader is

referred to [16], [17]. In Section II-B1 we formalize the problem of decoding motor intentions in BCIs as a non-parametric hypothesis testing problem. Then, in Section II-B2, we motivate the use of plug-in decoders based on minimax-optimal function estimators. The theory of Gaussian sequences is presented in Section II-B3 as a framework for devising tractable, asymptotically minimax-optimal function estimators. We then discuss the main result in Pinsker's theorem, as well as the blockwise variant of the James-Stein estimator in Sections II-B4 and II-B5, respectively. The connection between the two estimators, namely Pinsker's and the blockwise James-Stein is discussed in Section II-B6 where it is shown that in the context of the available LFP data, both estimators can be viewed as a special cases of the truncation estimator. We wrap up with Section II-B7 where we investigate the validity of the assumptions imposed by the theorems on the acquired data.

1) Problem Statement: Let the sequence Y_t for $t = 0, 1, \ldots, T-1$ represents a discrete-time LFP signal collected from an arbitrary electrode with sampling frequency $\nu_S = 1$ kHz during the memory period of an arbitrary trial (see Section II-A for a recap on the experimental setup). We assume that the LFP signal consists of two components [16], [17]:

1) information-carrying but unknown signal waveform, represented by a smooth function $f : [0, 1] \to \mathbb{R}$ that determines the dynamics of the decision-making process,
2) random noise component, σZ modelled as a zero-mean, independent and identically distributed (i.i.d.) Gaussian process with standard deviation σ.

These two components add up in the following, non-parametric, discrete-time model:

$$Y_t = f_t + \sigma Z_t, \quad Z_t \sim \mathcal{N}(0, 1), \quad t = 0, \ldots, T-1, \tag{1}$$

where $f_t = f(t\nu_S)$ and Z_t denote the discrete-time versions of f and Z, respectively. We assume that $f \in \mathcal{F}$. More precise technical assumptions about the function space $\mathcal{F}$ will be made later in Section II-B4; for now, it suffices to say that $\mathcal{F}$ is a class of smooth functions in the space of all square-integrable functions $\mathcal{L}_2([0, 1])$ with respect to a Lebesgue measure.

It is well-known that neuronal activity is non-stationary across time, space as well as subjects [35], [36]. This phenomenon occurs as a result of the natural variability of the recording conditions as the experiment proceeds; we use the term recording condition to encompass the various different factors that jointly determine the relationship between the activity of the neurons and the intended motor action, i.e., eye movement direction in our case [28]. Variations in the recording conditions might occur due to, for instance, changes in the electrical properties

of the measurement equipment including the electrode array, small drifts of the positions and depths of the electrodes, i.e., changes of the array topology as the experiment proceeds, psycho-physiological changes of the subject such as variations in the firing patterns of individual neurons, changes in the attention of the subject, etc. To summarize, it is reasonable to assume that each individual trial, conducted under the same experimental conditions will produce a different function f. We can formalize these notions as follows. First, we assume that the function f is *different* for each target location $k = 1, ..., K$ in the experiment (with $K = 8$ in our case, see Section II-A). Second, for a given target location, the function f *varies* across different trials. As a result, we conclude that each target location k forms a sub-class of functions $\mathcal{F}_k$ in the function space $\mathcal{F}$. In light of this, our objective can be expressed as a hypothesis testing problem; namely, we aim to find a decision rule $\delta(\cdot)$ that maps an arbitrary LFP sequence $Y_t, t = 0, ..., T-1$ into one of the $K = 8$ possible target classes, i.e., sub-classes of functions $\mathcal{F}_k, k = 1, ..., K$:

$$\hat{k} = \max_{k=1,...,K} \delta(\{Y_t\}_{t=0,...,T-1}).$$

2) Minimax-optimal function estimation: Next, we consider the *plug-in* decoder based on the function estimate of the true LFP signal waveform using the LFP sequence $\{Y_t\}$ and motivate the use of minimax-optimal function estimators. Let $\hat{f}(\{Y_t\}) = \hat{f}_T$ denote an estimator of f based on the time sequence $Y_t, t = 1, \ldots, T$ and the model (1). Consider the *plug-in decoder* $\delta(\{Y_t\}) = \delta(\hat{f}_T)$. A desired property of the plug-in decoder $\delta(\hat{f}_T)$ to be consistent, i.e., for high sampling rates the probability of miss-classification should go to zero asymptotically. One way to enforce and guarantee consistency is to take the worst-case miss-classification probability to zero as $T \to \infty$. It has been shown in [15] that a simple *minimum distance* decoder has a worst case miss-classification probability that goes to zero in the limit $T \to \infty$ if the function estimator $\hat{f}_T$ is consistent (in the sense that its mean squared estimation error goes to zero) as long as the function classes $\mathcal{F}_k, k = 1, \ldots, K$ are well-separated.

This motivates the use of minimax-optimal function estimators. Specifically, let $\hat{f}_T^*$ denote the estimator of f that minimizes the *maximum* mean squared estimation error over *all* function estimators $\hat{f}_T$ [16], [17]; in such case, it can be shown that the plug-in decoder $\delta(\hat{f}_T^*)$ converges fastest to zero among all classifiers of this type; furthermore, the consistency result is valid for any choice of function classes $\mathcal{F}_k, k = 1, \ldots, K$ as long as they remain well-separated. In other words, minimax-optimal estimators are robust with respect to the structure of the function space

$\mathcal{F}$; this is a desirable property, which should be retained given the non-stationary nature of the LFP signals.

3) Gaussian sequence model: Function estimators (including minimax-optimal estimators) are, in general, objects with infinite number of dimensions and difficult to use in practical BCIs. This suggests that we need to resort to finite-dimensional, asymptotically minimax-optimal representations of the minimax-optimal function estimators. The theory of Gaussian sequence models, which we present next, provides the necessary framework for designing such finite-dimensional estimators.

Consider a sequence $\{y_l\}$ of random variables, indexed in an index set $\mathcal{I}$. It is assumed that the components of the sequence $y_l, l \in \mathcal{I}$ are statistically independent and follow Gaussian distributions with unknown means θ_i and known positive and equal standard deviations ϵ. Thus, the sequence model can be written as

$$y_l = \theta_l + \epsilon z_l, \quad z_l \sim \mathcal{N}(0,1), \quad l \in \mathcal{I}. \tag{2}$$

In the above abstract model, θ_l are components that are not necessarily related to each other [16], [17]. We now show how to establish relationship between (2) and the regression model introduced in (1). Let us first define the vectors

$$Y = (Y_0, \ldots, Y_{T-1}), \quad \mathbf{f} = (f_0, \ldots, f_{T-1}), \quad Z = (Z_0, \ldots, Z_{T-1}).$$

These vectors live in T-dimensional Euclidean space, endowed with the normalized inner product $\langle \cdot, \cdot \rangle$ and the corresponding norm $\| \cdot \|_2$. Let the set of T-dimensional vectors $\{\varphi_l\}$ be an orthonormal basis with respect to a normalized inner product. Thus, we can form the following inner products:

$$y_l = \langle Y, \varphi_l \rangle, \quad \theta_l = \langle \mathbf{f}, \varphi_l \rangle, \quad z_l = \sqrt{T} \langle Z, \varphi_l \rangle,$$

for $l = 1, \ldots, T$. It is trivial to confirm that under the model (1), $z_l \sim \mathcal{N}(0,1)$; hence, the observation sequence $\{y_l\}$ satisfies the Gaussian sequence model (2) with $\epsilon = \sigma/\sqrt{T}$ and $\mathcal{I} = \{1, \ldots, T\}$. We conclude that the transition from the function space to the sequence space is done through an orthogonal linear transformation of the regression model (1) which leads to the model (2).

4) Pinsker's theorem: Instead of estimating f in the function space $\mathcal{F}$ using the regression model (1), we can alternatively estimate $\{\theta_l\}$ in the sequence space using the representation (2) with $\epsilon = \sigma/\sqrt{T}$. In light of this, Pinsker's theorem is a fundamental result in the theory of Gaussian sequences and a key ingredient in our deep learning architecture. Specifically, let Θ be in ellipsoid in $\ell_2(\mathbb{N})$ such that

$$\Theta(a, C) = \left\{ \theta = (\theta_1, \theta_2, \ldots) : \sum_l a_l^2 \theta_l^2 \leq C \right\},$$

where $\{a_l\}$ denotes a non-decreasing sequence of positive numbers. Let $\hat{\theta} = (\hat{\theta}_1, \hat{\theta}_2, \ldots)$ denote an estimator of $\theta \in \Theta$. Pinsker's theorem states that the estimator $\hat{\theta}^*$ of the form

$$\hat{\theta}_l^* = \underbrace{\left(1 - \frac{a_l}{\mu}\right)_+}_{c_l} y_l = c_l y_l, \quad \mu > 0, \tag{3}$$

is consistent and asymptotically minimax-optimal over the ellipsoid $\Theta(a, C)$. We see that Pinsker's estimator shrinks the observations by an amount $c_l = 1 - a_l/\mu$ as long as $a_l/\mu < 1$; otherwise, it attenuates them to zero. It is also linear, as the shrinking coefficient c_l does not depend on the observations. Finally, it is diagonal as the estimate $\hat{\theta}_l^*$ depends only on y_l and not on any other $y_j, j \neq l$.

One might wonder how is the above, rather abstract result related to our problem, namely classification of LFPs for decoding motor intentions. So far we have only showed how to transition from the function space representation (1) into the sequence domain representation (2) through an arbitrary linear transformation. However, we have still not discussed which transformation to use such that Pinsker's theorem is applicable.

Gaussian sequence representations obtained using the Fourier basis functions with θ constrained to live in an ellipsoid with a given sequence $\{a_l\}$, correspond to functions f living in function space $\mathcal{F}$ with a given structure [16], [17]. Specifically, consider the Fourier basis of sines and cosines:

$$\varphi_1(x) = 1, \quad \varphi_{2m}(x) = \sqrt{2}\cos(2\pi m x), \quad \varphi_{2m+1}(x) = \sqrt{2}\sin(2\pi m x), \quad m \in \mathbb{N}, \tag{4}$$

with $x \in [0, 1)$. Let Σ denote a Sobolev class of functions, defined as

$$\Sigma(\alpha, C) = \left\{ f \in \mathcal{L}_2([0,1]) : f^{(\alpha-1)} \text{ is absolutely continuous, and } \int_0^1 [f^{(\alpha)}(t)]^2 dt \leq C^2 \right\}.$$

It can be shown [16], [17] that a function is in the Sobolev function class $\Sigma(\alpha, \pi^\alpha C)$ if and only if its Fourier series coefficients $\{\theta_l\}$ are in an ellipsoid $\Theta(a, C)$ with

$$a_1 = 0, \quad a_{2m} = a_{2m+1} = (2m)^\alpha, \quad m \in \mathbb{N}.$$

We conclude that finding the (asymptotically) minimax-optimal estimator over a class of smooth functions $\Sigma(\alpha, \pi^\alpha C)$ is equivalent to finding the (asymptotically) minimax-optimal estimator over $\Theta(a, C)$. Thus, to obtain the minimax-optimal function estimator, one can take the Fourier series of the observations Y using the basis functions (4), with $x = \{t/T\}, t = 0, \ldots, T-1$, shrink the coefficients as in Pinsker's result (3), and then reconstruct a signal using the modified Fourier series coefficients. In addition, since $\{a_l\}$ is a monotonically increasing sequence, the sequence $\{c_l\}$ will be monotonically decreasing and the optimal estimator (3) has only a *finite* number of non-zero components. This is the key simplification provided by the sequence representation (2) and Pinsker's estimator (3); namely, it allows us to transition from the infinite-dimensional function space (i.e., the time domain (1)) to an equivalent finite-dimensional representation in the sequence space (i.e., the frequency domain) where only a handful of Fourier coefficients corresponding to the lowest frequencies are non-zero. In other words, Pinsker's theorem tells us that the optimal regressor for functions assumed to exhibit Sobolev type of smoothness is weighted low-pass filtering.

5) The blockwise James-Stein variant: Pinsker's estimator (3) depends on the Sobolev smoothness parameters α and C, or, equivalently α and μ, which in practice are not known. In other words, these parameters should be treated as hyper-parameters, meaning that they should be optimized using the training data via some form of cross-validation. However, in setups with limited training data, this might lead to overfitting; in other words, finding the adequate Sobolev class parameters from limited data is not a trivial task. An alternative approach that does not require knowledge of the smoothness parameters is the James-Stein estimator and its blockwise extension, discussed next.

Let $y = (y_1, \ldots, y_n) \sim \mathcal{N}(\theta, \epsilon^2 I)$, where $\theta = (\theta_1, \ldots, \theta_n)$ be a multivariate diagonal Gaussian random vector. Consider the problem of estimating θ from y. The James-Stein estimator of the vector θ, defined for $n \geq 2$, is given by [16]:

$$\hat{\theta}_{\text{JS}}(y) = \left(1 - \frac{(n-2)\epsilon^2}{\|y\|_2^2}\right)_+ y. \tag{5}$$

The above nonlinear estimator is uniformly better than the maximum likelihood estimator y for this problem [16]. We observe that unlike Pinsker's estimator (3), the James-Stein estimator is

not linear, as the amount of shrinkage applied to the observations y depends on the norm of the observations. Also note that the amount of shrinkage applied is equal for all observations; hence, the James-Stein estimator is still diagonal.

For the Gaussian sequence model (2), a blockwise variant of the James-Stein estimator is defined [16], [17]. We first divide the Gaussian sequence $\{y_l\}$ into blocks. Then we apply the James-Stein estimator (5) for each block. Consider a partition of the positive integers $\mathbb{N} = \cup_{j=0}^{\infty} B_j$, where $B_j = \{2^j, \ldots, 2^{j+1} - 1\}$ are disjoint dyadic blocks; the size of block j is $|B_j| = 2^j$. We define the observation vector for block j as $y^{(j)} = \{y_l, l \in B_j\}$. We fix the integers L and J. Then, the blockwise James-Stein estimator is defined as

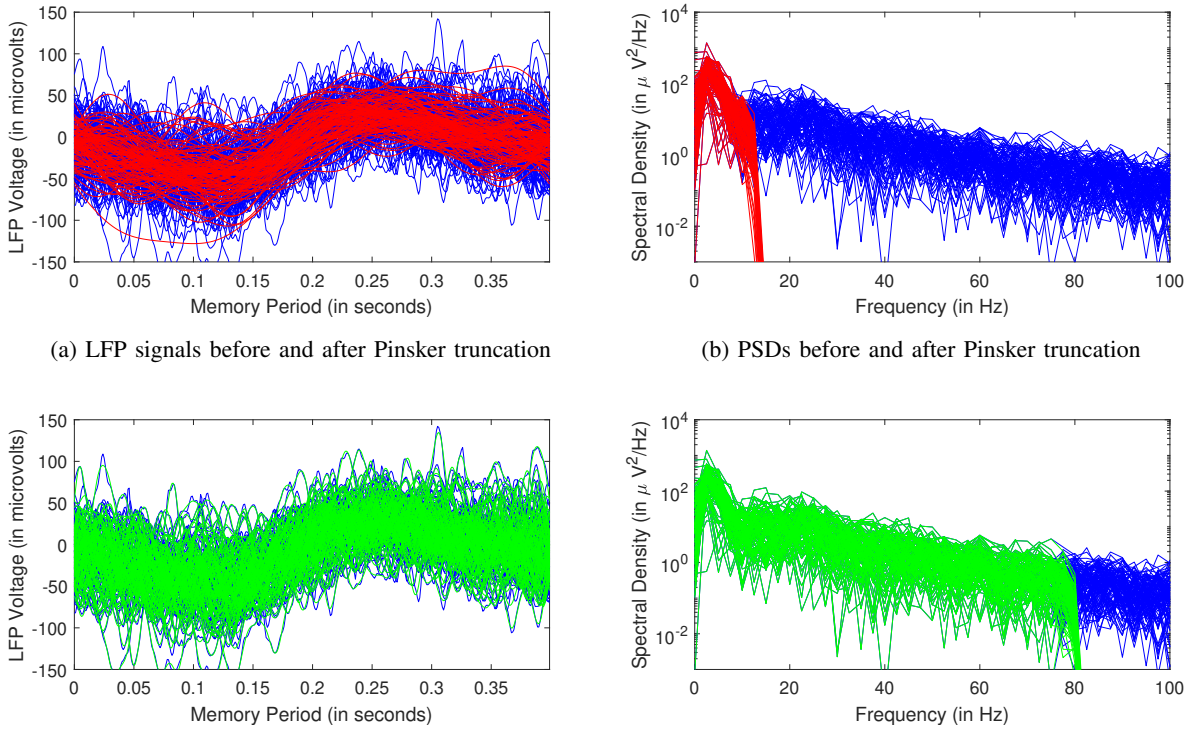
$$\hat{\theta}^* = \begin{cases} y^{(j)} & j \leq L \\ \hat{\theta}_{\text{JS}}(y^{(j)}) & L < j \leq J \\ 0 & j > J \end{cases}, \tag{6}$$

where, $\hat{\theta}_{\text{JS}}(y^{(j)})$ is computed using (5) with $n = 2^j$. The blockwise James-Stein estimator leaves the first L blocks unchanged (i.e., $c_l = 1$ for $l \in \{1, \ldots, 2^L - 1\}$), applies James-Stein shrinkage to the next $J - L$ blocks (i.e., c_l is computed according to (5) for $l \in \{2^L, \ldots, 2^J - 1\}$), and shrinks the rest of the observations to 0 (i.e., $c_l = 0$ for $l \geq 2^J$). The integer J is typically chosen to be of the order of $\log \epsilon^{-2}$. Considering (2), we see that $\epsilon \sim 1/\sqrt{T}$; thus, we can fix $J = \log T$ and the only free parameter is L. The blockwise James-Stein estimator (6) is known to adapt asymptotically to the convergence rate of the minimax-optimal estimator for each dyadic ellipsoid [16]. Namely, if θ satisfies $\sum_{j \geq 0} 2^{2j\alpha} \sum_{l \in B_j} \theta_l^2 \leq C^\alpha$, where $\alpha, C > 0$ are the ellipsoid parameters, then (6) automatically adapts the convergence rate of the minimax estimator for each α and C without using the ellipsoid parameters in its design; this property is known as adaptive minimaxity [16], [17]. It should be noted however, that adaptive minimaxity is an asymptotic property valid only in the limit $T \to \infty$.

6) The truncation estimator: The truncation estimator of the form

$$\hat{\theta}^* = \text{diag}(1_{l \leq Q}) y, \tag{7}$$

is also asymptotically minimax-optimal [16]. Here, $1_{l \leq Q}$ is a vector where the first $Q \leq T$ entries are 1 and the remaining 0; hence, the truncation estimator can be viewed as a special case of Pinsker's estimator (3) with $c_l = 1$ for $l \leq Q$ and $c_l = 0$ for $l > Q$, corresponding to a Sobolev class of infinitely-differentiable functions, i.e., $\alpha \to \infty$ and parameter $\mu \to \infty$; in other words, the finite-dimensional representation of the estimate of f is obtained by simply retaining



(a) LFP signals before and after Pinsker truncation

(b) PSDs before and after Pinsker truncation

(c) LFP signals before and after James-Stein filtering

(d) PSDs before and after James-Stein filtering

Fig. 3. The LFP signals and their spectral densities. An arbitrary channel in Monkey A's EDC-6 data set and the eye movement direction "←" were arbitrarily selected; same conclusions apply to any other channel, movement direction, data set and subject. Left-hand plots: blue lines correspond to original LFP sequences with $T = 400$, red lines on top plot correspond to low-pass filtered waveforms using (7) with $Q = 9$, green lines on bottom plot correspond to low-pass filtrered waveforms using (6) with $L = 2$ and $J = \lfloor \log T \rfloor = 5$. Right-hand plots: Spectral densities of the original (blue lines) and filtered LFPs (red and green lines, respectively). The PSD was estimated using Welch's method [37] with Hann window, segment size of $T/2$ and overlap of 50%.

the first Q Fourier coefficients, corresponding to the $(Q+1)/2$ dominant frequency components of the complex spectrum of the LFP signal. The main difference between Pinsker's (3) and the truncation estimator (7) arises in their respective rates of convergence; namely, Pinsker's estimator converges fastest to the true LFP waveform as $T \to \infty$ among all minimax-optimal estimators [16]. In practice, this is a rather subtle difference and one should not expect significant deviation in the decoding performance between (3) and (7), which our experiments in the past have confirmed [14]; namely, the performances of a plug-in decoders based on (3) and (7) remain virtually the same [14]. The truncation estimator is also simpler to implement than Pinsker's estimator since it introduces only a single free parameter, namely the number of retained Fourier

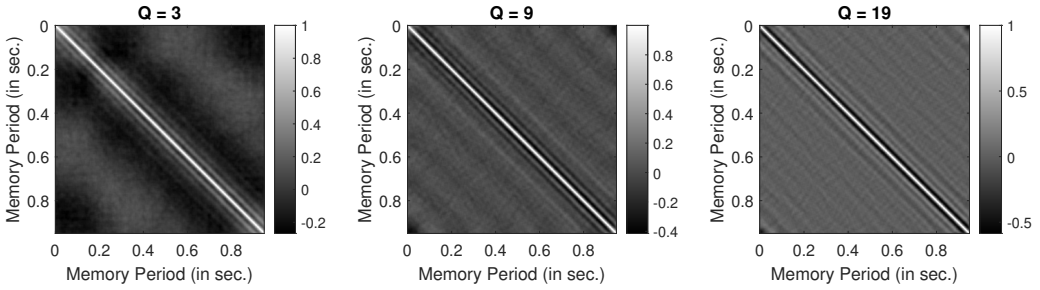
coefficients Q, corresponding to a cut-off frequency of

$$\frac{(Q-1)\cdot\nu_S}{2T}.$$

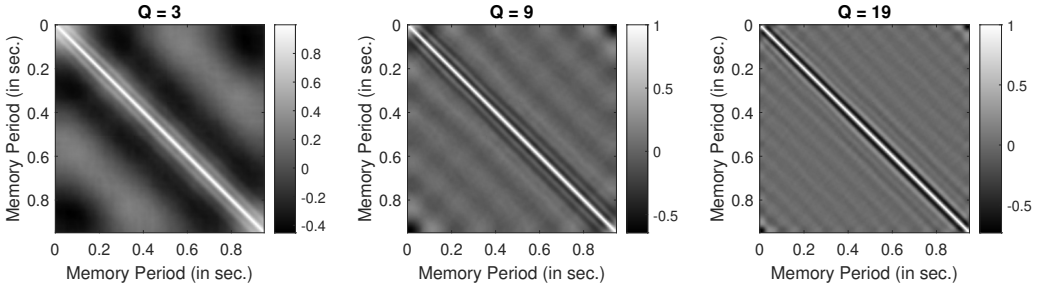
One should note that the parameter Q is an *odd* number, see also (4); one real number is stored for the DC component of the signal, and two real numbers are stored for each retained frequency, namely the sine and cosine of the Cartesian representation of the complex Fourier coefficient corresponding to each frequency component.

Pinsker's (3), the blockwise James-Stein variant (6) and the truncation estimator (7) are all quite similar. In essence, all three act as low-pass filters, suggesting that as long as the LFP data meets the smoothness criteria, one should only consider the lowest frequency components and attribute higher frequency components to the noise. The number of retained Fourier coefficients Q is, however, specific for each of the estimators; for instance, in Pinsker's estimator (3), Q is a function of the smoothness parameters of the Sobolev class α and the parameter μ, whereas in the blockwise James-Stein variant, Q is a function of the integers L and J with the latter being determined by T as discussed in Section II-B5. Moreover, from (5), it is not immediately clear how does the sequence of shrinkage coefficients $\{c_l\}$ behave for $l \in \left\{2^L, \ldots, 2^J - 1\right\}$ since their values depend on the actual data; in other words, their values might vary non-monotonically between 0 and 1. Fig. 3 shows exemplary LFPs and their Pinsker/James-Stein filtered versions in the time and frequency domains (see caption for more details). By comparing the PSD profiles depicted in Figs. 3b and 3d, we see that the block-wise James-Stein estimator effectively behaves like the truncation estimator (7) with $Q = \sum_{j=0}^{J} 2^j$. We conclude that, for typical values of T the shrinkage coefficients from (5) are $c_l \approx 1$ for $l \in \left\{2^L, \ldots, 2^J - 1\right\}$; as a result, the blockwise James-Stein method (6) tends to produce more variable LFP waveform estimates by retaining larger number of the higher frequency components.

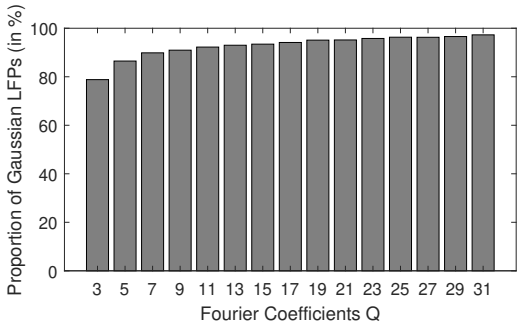
7) Statistical analysis of LFP signals: The successful application of the non-parametric regression framework to LFP data depends crucially on the validity of the model (1). To assess the validity of this model, we analyse the statistical properties of LFPs. To supress the noise from the LFP data, we first low-pass filter the LFPs using the truncation estimator (7), and then we subtract the reconstructed, smooth signal from the original signal. We then compute the correlation matrix and quantify the Gaussianity of the LFP noise using the standard Kolmogorov-Smirnov goodness-of-fit (KS-GOF) test [38]. The null hypothesis in single-trial KS-GOF test is that the LFP noise values are independent realizations of the standard normal distribution;



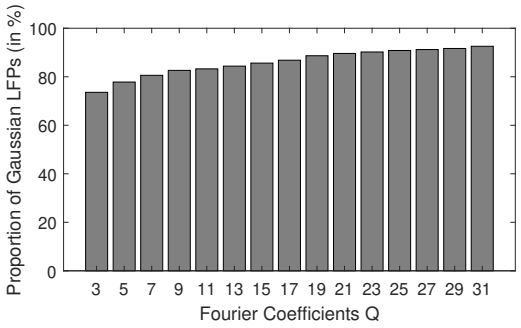
(a) Sample correlation matrices for varying Q: Monkey A



(b) Sample correlation matrices for varying Q: Monkey S



(c) Kolmogorov-Smirnov normality test: Monkey A



(d) Kolmogorov-Smirnov normality test: Monkey S

Fig. 4. Statistical analysis of the LFP noise. An arbitrary channel was selected for illustration; the conclusions are consistent across channels. Top and middle row plots: correlation matrices of the noise component, averaged over all trials. The original LFP signals were low-pass filtered with (7). Bottom row plots: proportion of noise sequences that confirmed the null hypothesis in Kolmogorov-Smirnov goodness-of-fit test with significance level of 0.01.

therefore, before running the test, we normalize the noise sequence by dividing with its standard deviation. Note that the noise sequence is already zero-centered due to the minimax-optimality of the truncation estimator (7), which guarantees it to be an unbiased estimator of the true signal waveform [16]. The results are shown in Fig. 4. The top and middle rows depicts the correlation matrices of the noise; we can clearly see that the matrices for both subjects are diagonally dominated, confirming the adequacy of the temporal independence assumption from (1). The bottom row plots show the proportion of all LFP noise sequences that have confirmed the null

hypothesis with significance level of 0.01 for multiple values of Q in the range from 3 to 31. We observe that the normality assumptions required in the model (1) is also adequate; even in the case of $Q = 3$, which corresponds to 2 retained components (i.e., the DC and the lowest frequency component), the proportion of LFP noise sequences that confirmed the null hypothesis is nearly 80% in both subjects.

C. Deep Learning Architecture with Non-parametric Feature Extraction

Our deep learning architecture, schematically depicted in Fig. 5, is a time series classification algorithm where features are extracted from raw LFP recordings using the non-parametric function estimators presented in Section II, while the decoding rule $\delta(\cdot)$ is a DNN. The LFP data is collected via multielectrode array with N channels during the memory period of each trial; as indicated earlier in Section II-A, we begin sampling and recording the LFP signals at the beginning of the memory period, immediately after the target light is extinguished. The number of LFP samples T per electrode (which we also refer to as sampling window) is a hyper-parameter and should be chosen carefully; this is motivated by the fact that not all samples during the memory period encode equally relevant information. Earlier analyses [14] have shown that the early time epochs of the memory period determine the dynamics of the decision-making process with the majority of information relevant for decoding stored within the first half of the memory period. After acquiring the LFP signals from each channel, we apply a non-parametric function estimator; namely, we use either Pinsker's estimator (3) or the blockwise James-Stein estimator given in (6) to compute and adequately shrink Q non-zero Fourier coefficients, corresponding to the real and the imaginary part of the Cartesian representation of the lowest $(Q + 1)/2$ frequency components (including the DC component) for each channel individually, which are then concatenated into single, large feature vector. Finally, we pass the feature vector through a fully-connected MLP with K outputs and softmax output activation function and use $\max(\cdot)$ over the output vector to determine the intended motor action of the subject.

To summarize (see Fig. 5):

1) Fix the hyper-parameters: sampling window T, number of Fourier coefficients per channel Q (see below) and the DNN parameters (number of hidden layers, number of hidden neurons per layer, hidden neuron activation functions and optimization algorithm parameters such as learning rate, minibatch size and number of training epochs).

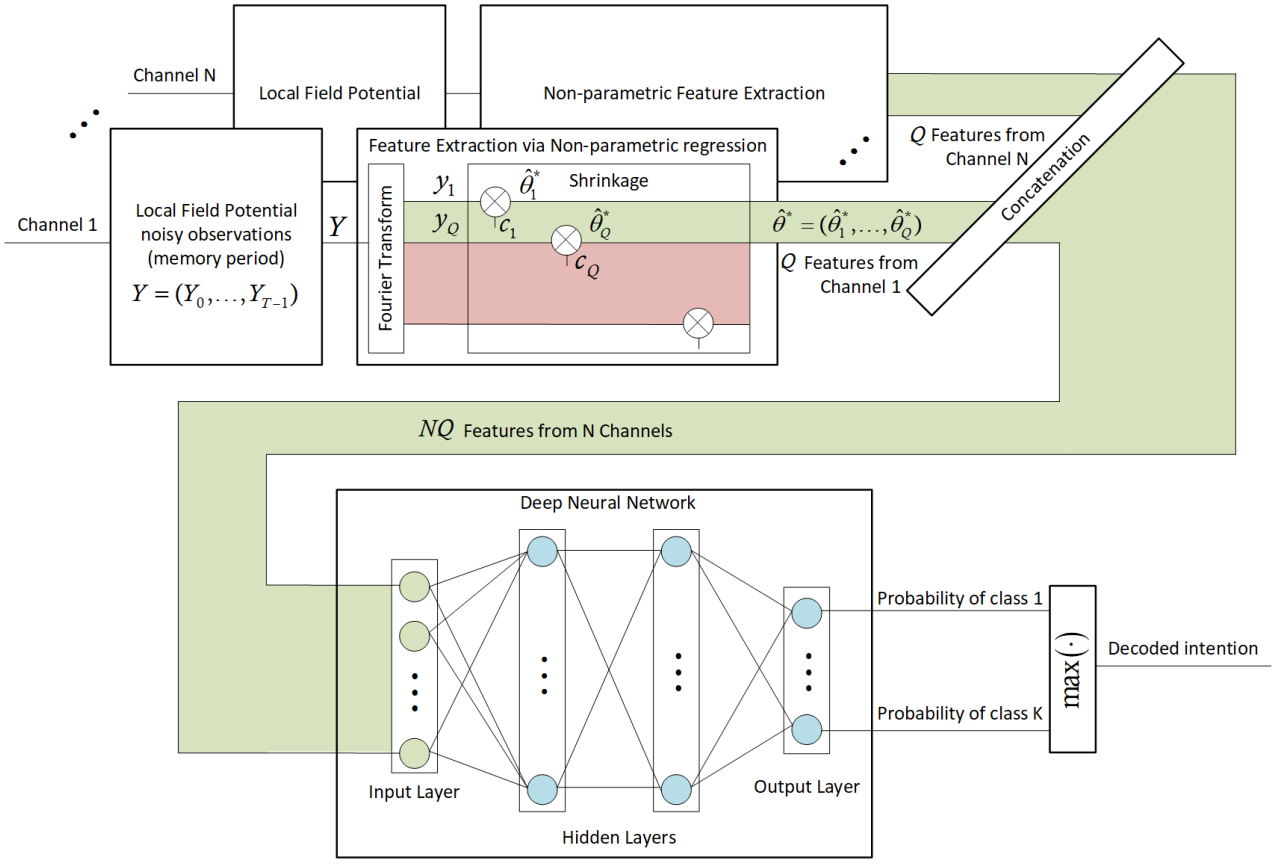


Fig. 5. The deep neural network with non-parametric feature extraction. See Section II-C for details.

2) *Data acquisition.* Collect N LFP sequences of length T samples from the memory period of each trial.

3) *Non-parametric feature extraction.* Compute Q real Fourier coefficients per LFP sequence, using the basis functions (4) and apply shrinkage following Pinsker or blockwise James-Stein.

- *Deep Pinsker Neural Network (DPNN).* Features are extracted using Pinsker's method (3). In this case, Q is a function of the parameters of the Sobolev space, or, equivalently, the ellipsoid parameters. Specifically, we first fix the smoothness parameter α of the Sobolev space and the free parameter μ. The shrinkage coefficients are computed via (3) with the number of non-zero Fourier coefficients corresponding to the largest number $Q = Q(\alpha, \mu)$ such that $c_Q > 0$ and $c_{Q+1} = 0$; note that the monotonically increasing sequence $\{a_l\}$ ensures the existence of such Q for any value of the parameter μ. As an

alternative, within the framework of the DPNN we can also apply the truncation estimator (7) that has only one free parameter, namely the number of retained Fourier coefficients Q; our investigations [14] show that for fixed number of Fourier coefficients Q, there is virtually no difference in performance between the Pinsker's method and the truncation estimator.

- *Deep James-Stein Neural Network (DJSNN).* Features are extracted using the blockwise James-Stein estimator (6). We fix the free integer $L < J$ where $J = \lfloor \log T \rfloor$ and we apply the shrinkage according to (6); the number of Fourier coefficients in this case is $Q = Q(J) = \sum_{j=0}^{J} 2^j$.

4) *Feature space formation.* The Fourier coefficients from each channel are concatenated to form a single high-dimensional feature vector of length NQ; this is the dimension of the feature space.
5) *(Optional) Dimensionality reduction and/or standardization.* Use principal component analysis and/or feature standardization (alternatively whitening) to reduce the dimension of the feature space.
6) *Decoding motor intentions.* Pass the high-dimensional feature vector through a pre-trained DNN and decode the intended action.

The performances of the DPNN and the DJSNN are optimized over the hyper-parameters using cross-validation. Special attention needs to be devoted when selecting the values for the hyper-parameters of the DNN to avoid overfitting. This is especially important in BCIs and similar neural engineering applications since here the number of available data points is often quite limited (in many cases severely so). Additionally, the parameters related to the optimization procedure used to train the network such as learning rate and minibatch size, including the choice of optimization algorithm itself are also a subject to fine-tuning via cross-validation.

III. Results

In this section we present our main findings. In Section III-A, we present the hyper-parameter optimization procedure where we focus on finding the best-performing MLP configuration. Section III-B presents the results from the evaluations.

A. Hyper-parameter Optimization

As discussed in Section II-A3 (also pictorially depicted in Fig. 2), the average number of trials per EDC in Monkey A and Monkey S is ≈ 90 and 250, respectively. Given the dimension of the feature space which easily surpasses 100 (even after dimensionality reduction, see [14], [15]) and can go up to several hundreds, we conclude that the number of trials available per EDC might be (in most cases) insufficient for reliable training of the DNN. To obtain larger data sets, we use a data bundling algorithm, similar to the one used in [14]. The algorithm is based on the Euclidean proximity of the EDC vectors and uses the following reasoning: EDCs whose depth vectors are close in Euclidean sense, generate similar feature spaces with similar target-conditional distributions. The algorithm operates as follows. First, we define the clustering window as the minimum number of trials per EDC data set, which we also refer to as an EDC data cluster. Second, we fix the concurrent EDC at which we want to create data set of larger size. Next, we begin appending trials for the concurrent EDC; we first append the trial from the concurrent EDC and if the number of trials is less than than the clustering window, we start appending trials from the EDCs which are closest to the concurrent EDC in Euclidean sense. The algorithm ends when the total number of trials fills up the clustering window. For more details, the reader is referred to [14] where the procedure has been first reported and applied over the same data. It should be noted that the process of clustering data trials collected across different EDCs further increases the possibility of function class overlap in the function space. Recall from the discussion in Section II-B1 that the recording conditions vary as the experiment progresses, leading to the displacement of function classes across the multidimensional feature space. Grouping multiple small data sets, each of them collected for fixed but (even slightly) different EDCs will eventually lead to an overlap between function classes associated with different eye movements. Therefore, the clustering window becomes an additional hyper-parameter whose value should be chosen carefully in a manner that strikes the optimal trade-off between reliability of the trained DNN and the performance degradation that might occur due to increased function class overlap.

The hyper-parameters used in the experiments are summarized in Table I. In our evaluations, we primarily focus on optimizing the hyper-parameters related to the DNN. In other words, we did not fine tune the performance over the parameters of the feature extraction methods; rather, we fixed these parameters to reasonable values which are suggested in prior works [14]. For

TABLE I

Hyper-parameter	Value
Feature Extractors	
Number of LFP samples	$T = 650$ (first 0.65 seconds of memory period) [14]
Pinsker/truncation: number of retained Fourier coefficients	$Q = 9$
Blockwise James-Stein integers	$L = 2$, $J = \lfloor \log T \rfloor = 6$
Number of principal modes (applies only to DJSNN)	$P = 185$
The Deep Neural Network Architecture	
Topology	fully-connected
Number of hidden layers	3
Number of neurons in hidden layers	64 (1st layer), 32 (2nd layer), 16 (3rd layer)
Hidden neuron activation functions	ReLU
Optimization method	Adam
Initial learning rate	0.001
Minibatch size	50
Number of training epochs	150

instance we fixed T to 650, corresponding roughly to the first half of the memory period. For Pinsker's feature extraction, we retain only $Q = 9$ Fourier coefficient per channel, corresponding to the first 5 lowest frequencies (including the DC component); equivalently, this corresponds to a cut-off frequency of 7 Hz. For James-Stein feature extraction, we fix $L = 2$, whereas $J = 6$, retaining (at most) $Q = 127$ non-zero Fourier coefficients per channel and corresponding to a cut-off frequency of 95 Hz.

We use the leave-one-out cross-validation (LOO-CV) method to fine-tune the remaining parameters and find a DNN configuration that offers the best decoding performance [33]; next, we describe how we do this in more details. First, we take a single data point out: this data point is our held-out set used to test the performance. We use the remaining data points from the data set to train the DNN. We search in the space of fully-connected MLPs. The size of the input and output layer is $32 \cdot Q$ and $K = 8$, respectively; the output layer applied softmax activation function. We fix the number of hidden layers, number of hidden units per layer, hidden unit activation functions, number of training epochs, size of the minibatch and the learning rate; we train the DNN using Adam optimization method [39] and record the training accuracy. We then test the trained DNN using the held-out set and record the outcome; namely, we record 1 if the

data point is correctly classified and 0 otherwise. We repeat the procedure for every single point from the data set and record the respective outcomes. At the end, we compute the average of both the training and the test accuracy (i.e., the number of 1s divided by the number of data points in the set); note that these averages correspond to the cross-validation risk [33].

We repeated the above LOO-CV procedure for multiple configurations of fully-connected DNNs and recorded the average training and test accuracy. Specifically, we vary the number of hidden layers ranging from 1 to 5, we investigate several different configurations for number of hidden units ranging from 8 to 500 (with step 50), we vary the number of training epochs ranging from 50 to 350 (with step 50), as well as minibatch sizes ranging from 15 to 400 (with step 25). For the learning rate, we used an adjustment technique that reduces the learning rate by a factor of 0.1 every 50 training epochs and we varied the initial learning rate as a hyper-parameter. Since there are multiple data sets, collected across different cortical depths and over multiple recording sessions spread out through multiple days, we run the LOO-CV method for several representative EDC data sets in parallel (selected from different cortical depths) and record the corresponding test performances. Although it is likely that different data sets will hint on different optimal DNN configurations, the variations are expected to be subtle; that is, our main goal is to find a DNN configuration that performs consistently well across *all* data sets, *all* cortical depth and *all* subjects. We were able to find such configuration; its parameters are summarized in Table I and its performance in terms of the average test accuracy is presented in the following Section III-B. We note that the average training accuracy for each experiment ranged from 95% to 100%; the average test accuracy, on the other hand, varied noticeably across different configurations. As shown in Table I, a relatively shallow network with only 3 hidden layers and relatively small number of hidden, rectified linear units (ReLU) produced the best results; having a small DNN in our experiments is expected since the number of trials is quite limited with larger networks showing tendency to overfit. Specifically, we have observed that the average test accuracy tends to drop considerably as we increase the number of hidden layers beyond 3. We have also observed that the average test accuracy remains more stable as we increase the number of hidden units per layer for a fixed number of layers; however, after increasing the number of units per hidden layer beyond 250, the DNN is heavily over-parametrized, the training accuracy is 100% and the test accuracy drops considerably. The initial learning rate also had a strong impact: for instance, initial learning rates larger than 0.001 led to a dramatic drop in the test accuracy.

B. Evaluations

We begin by comparing the performance of the DPNN with the state-of-art performance achieved by applying an LDA classifier over Pinkser's feature space, denoted as Pinsker-LDA [14], [15]; instead of the exact Pinsker formula (3), we apply the truncation estimator (7). Also, we did not apply any dimensionality reduction technique or feature standardization procedure when evaluating the DPNN. The values of the hyper-parameters are given in Table I. At first, we keep the data sets unchanged, i.e., we do not perform data clustering and we apply the decoding methods directly over the data sets with respective sizes depicted in Fig. 2. The results are shown in Fig. 6 where the test (i.e., decoding) accuracy is presented against the mean electrode depth of the corresponding EDC (the same metric is used in Fig. 2). We immediately see that even though the number of trials is quite small across all data sets, the DPNN outperforms the LDA-based method. We attribute this behaviour to the robustness of the Pinsker feature extraction method and its ability to extract meaningful features from LFP signals; this allows the DNN to be reliably trained even in a limited data setup. This is further implied when we investigated the performance of the DNN (with the same configuration parameters) applied directly over the LFPs where we observed significant degradation of performance. We note that the above results and observations are all along the line of common intuitions in time series classification problems; specifically, it is well known in time series classification that the feature extraction procedure is the crucial component that largely determines the performance of the system. Our results further confirm this while showing that feature extractors based on non-parametric function estimators are crucial for reliable training of a non-linear MLP classifier from limited data.

The results in Fig. 6 have shown that the deep non-parametric neural network achieves improvement over LDA-based method applied over the same features; nevertheless, the performance is still quite limited for any practical purpose. This is due to the size of the training data sets. In Fig. 6a, we observe that the performance of EDC-6 in Monkey A is substantially higher than the performance over the remaining data sets. This is not surprising, as we can see from Fig. 2a that EDC-6 is an exception in terms of the size of the data set; in addition, EDC-6 was acquired at superficial cortical depths, near the surface of the PFC, which is also an important factor in the resulting performance [12], [14].

To further improve the performance, we apply the data clustering method described in Section III-A. We investigated clustering windows with various widths, ranging from 200 to 2000

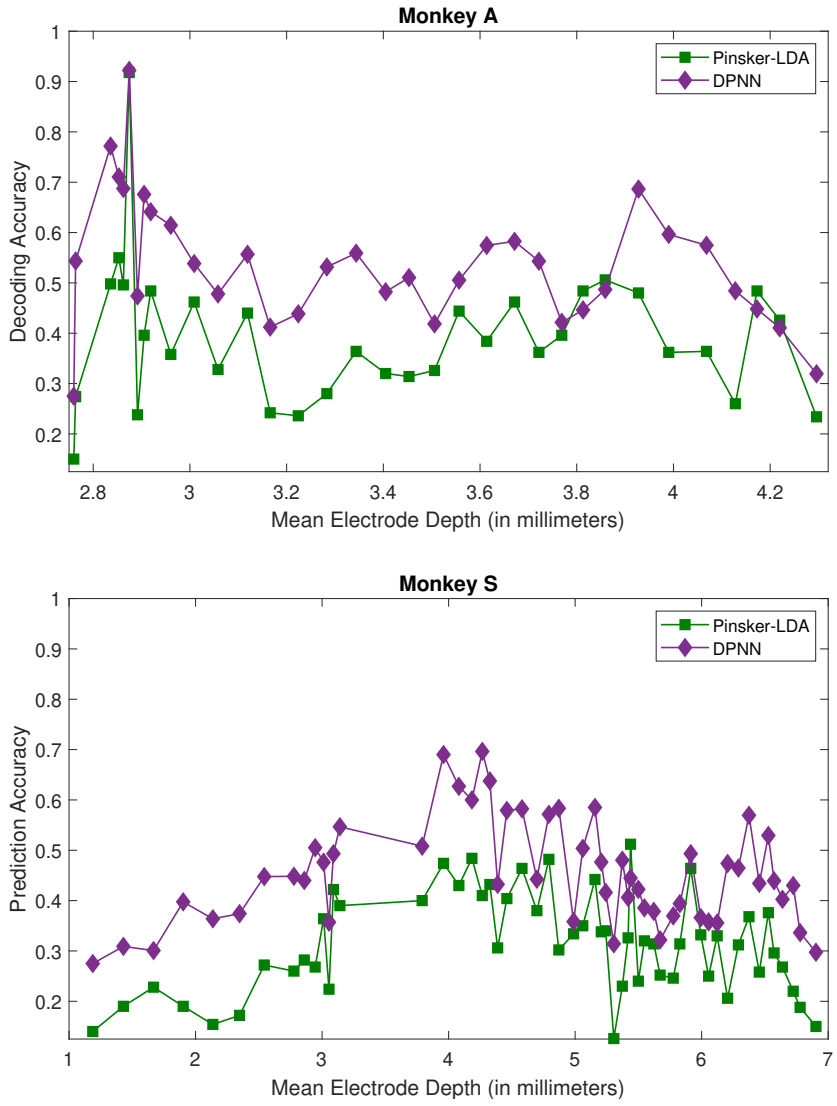


Fig. 6. The DPNN vs Pinsker-LDA method. The LDA classifier is applied over the same features as the DNN in the DPNN. Both methods are applied directly over the initial data sets, without data clustering. Remaining hyper-parameters are given in Table I.

for Monkey A and from 350 to 2000 for Monkey S. The best performance was achieved for a clustering window size of around 1000 trials with larger EDC data clusters showing a tendency of decrease in the decoding accuracy. The results corresponding to clustering window of 1000 trials are shown in Fig. 7; note that we investigated 5 EDC data clusters in total for Monkey A (namely, the data clusters formed around EDC-6,12,18,24 and 30) and 9 EDC data clusters for Monkey S (i.e., EDC-6,12,18,24,30,36,42,48 and 54). We also compare the performance of the DPNN against the performance of the LDA applied over both the Pinsker features and the

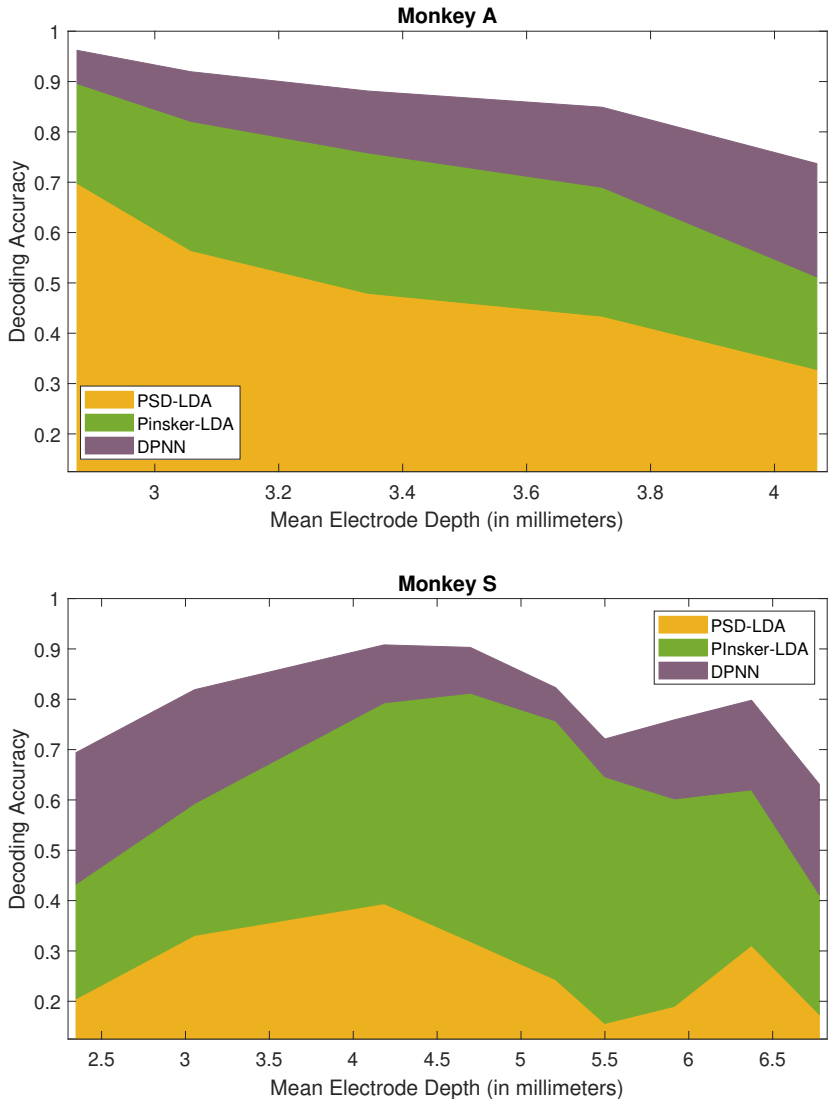


Fig. 7. The DPNN vs LDA-based methods. Clustering window size is 1000. Data clusters are formed around EDC-6,12,18,24 and 30 for Monkey A and EDC-6,12,18,24,30,36,42,48 and 54 for Monkey S. Remaining hyper-parameters are given in Table I.

more traditional PSD-based features where we used the absolute value of the complex Fourier coefficients. Evidently, the DPNN outperforms the LDA-based methods, peaking just below 97% and 90% decoding accuracy in Monkey A and S, respectively, as opposed to 90% and 80% when using LDA for the exact same EDCs over the exact same features. We note that such performance results have so far not been reported for the memory-guided visual saccades experiment; we attribute the remarkable performance gain to the deep learning component in the DPNN. Moreover, the peak performance in both subjects occurs at superficial cortical depths

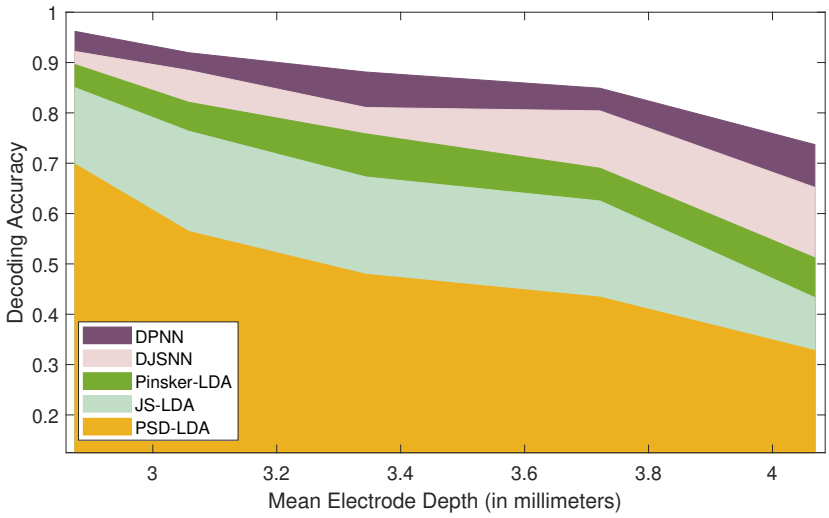


Fig. 8. The DJSNN vs DPNN and LDA-based methods over Monkey A data. Clustering window size is 1000. Data clusters are formed around EDC-6,12,18,24 and 30. Remaining hyper-parameters are given in Table I.

at sites near the surface of the prefrontal cortex, which is consistent with previous findings in the literature [12], [14]. We also see that the superior performance of DPNN over the LDA-based methods remains consistent across cortical depths. The findings are even more interesting at deeper cortical sites. The LDA-based analyses, which are widespread in the literature on eye movement decoding [12], [14], have consistently reported poor decoding performance when approaching white matter, suggesting that at deeper cortical sites the signal-to-noise ratio of the LFP signals is substantially larger than at superficial sites. In other words, the neuronal activity close to white matter does not produce enough informative signal for a BCI to produce reliable predictions. However, the results presented in Fig. 7 suggest that this is only partially true and part of the reason for the poor performance at deeper cortical sites are the inherent limitations in the classifier itself. As known, LDA is well suited for feature spaces where the target conditional distributions are Gaussian with the equal covariance matrices and high SNR [23], [24], which is not an entirely adequate assumption for our data. In this sense, using non-linear approach based on DNN alleviates part of the issue, peaking just above 70% at at deep cortical sites, which marks a substantial improvement over LDA.

Our final set of investigations concern the performance of the DJSNN; to this end, we apply the DJSNN with the respective parameters given in Table I over the same EDC-data clusters as in Fig. 7a. As already pointed out in Section III-A, with $T = 650$ we have $J = 6$ and $Q = 127$.

Thus, the dimension of the feature space after concatenating features across channels will be $32 \cdot 127 = 4064$, which is about 4 times larger than the number of trials in the data EDC data clusters with clustering window size of 1000. Therefore, in these analyses, we apply Principal Component Analysis (PCA) to reduce the dimension of the feature space; the optimal value for the number of principal modes is given in Table I. The results are shown in Fig. 8 where they are plotted together with the results from Fig. 7a for comparison. We see that the performance of the DJSNN consistently outperforms the methods based on LDA, but it remains inferior to the DPNN. This is an expected result since the behavior of the blockwise James-Stein estimator (6) is practically equivalent to the truncation estimator (7) for the LFP data as discussed in Section II-B6, albeit with (significantly) larger number of features per channel. On the other hand, it is well known from the empirical results in earlier works [14] that the most informative frequencies for decoding motor intentions from memory-guided visual tasks are stored in the delta band, i.e., below 5 Hz; hence, increasing Q (thereby increasing the number of dominant frequencies retained) leads to degradation of performance as many of the higher frequencies only bring additional noise.

IV. Discussion

We proposed a novel deep learning architecture for decoding intended motor actions in BCIs from limited data. The method combines famous results from the theory of non-parametric regression for feature extraction from LFP signals and DNN for classification. In this sense our deep neural networks bring the best of both the worlds of non-parametric regression and deep learning. Namely, the strong representational power of non-parametric function estimators and their robust and agile ability to extract meaningful and relevant features from time sequences allows the DNN to be trained with limited data, thus bridging the gap created by the scarcity of the data, and fostering the adoption of deep learning in neural engineering applications on a larger scale.

We considered two somewhat related architectures. The first one uses the famous Pinsker's estimator for feature extraction (i.e. the DPNN architecture), whereas the second one applies the blockwise version of the popular James-Stein estimator (leading to the DJSNN architecture). We tested the DPNN for decoding eye movement goals in memory-driven visual saccades to one of eight target locations on a data collected from two subject across various cortical depths. The results showed dramatic improvement of decoding accuracy over LDA-based methods,

demonstrating the exceptional capabilities of DPNNs. In particular, for Monkey A, the DPNN peaked at 96.5% at superficial cortical sites, near the surface of the PFC, whereas for Monkey S peaked at 90% again, at superficial cortical depth. Compared to LDA applied over the same features, this corresponds to a relative improvement of 7% for Monkey A and 12.5% for Monkey S at the same depths. Similarly, when compared to the more traditional PSD-LDA decoders that utilise the absolute Fourier coefficients as features, the DPNN shows relative improvement of 40% for Monkey A and 100% for Monkey S at superficial cortical depths. Even more remarkable, the performance of the DPNN at deep cortical sites peaked at 72% in Monkey A and 62% in Monkey S, marking a relative improvements of 38% (Monkey A) and 48% (Monkey S) over Pinsker-LDA and 125% (Monkey A) and 342% (Monkey S) over PSD-LDA. We also showed that the DJSNN architecture leads to a consistently inferior performance compared to the DPNN architecture; however, the DJSNN still outperformed the LDA-based methods, confirming the promising potential of the deep neural networks with non-parametric feature extraction for designing highly reliable BCI systems from limited training data.

ACKNOWLEDGMENT

This work was supported by the Army Research Office MURI Contract Number W911NF-16-1-0368.

REFERENCES

[1] K. Moxon and G. Foffani, "Brain-machine interfaces beyond neuroprosthetics," *Neuron*, vol. 86, no. 1, pp. 55 – 67, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0896627315002603

[2] R. P. N. Rao, *Brain-Computer Interfacing: An Introduction*. Cambridge University Press, 2013.

[3] S. J. Bensmaia and L. E. Miller, "Restoring sensorimotor function through intracortical interfaces: progress and looming challenges." *Nature Neuroscience*, vol. 15, no. 8, pp. 313–325, May 2014.

[4] J. M. Carmena, M. A. Lebedev, R. E. Crist, J. E. O'Doherty, D. M. Santucci, D. F. Dimitrov, P. G. Patil, C. S. Henriquez, and M. A. L. Nicolelis, "Learning to Control a Brain–Machine Interface for Reaching and Grasping by Primates," *PLoS Biology*, vol. 1, no. 2, p. e42, Oct. 2003.

[5] B. Pesaran, J. S. Pezaris, M. Sahani, P. P. Mitra, and R. A. Andersen, "Temporal structure in neuronal activity during working memory in macaque parietal cortex." *Nature Neuroscience*, vol. 5, no. 8, pp. 805–811, Aug. 2002.

[6] V. Gilja, P. Nuyujukian, and C. A. Chestek, "A high-performance neural prosthesis enabled by control algorithm design," *Nature*, 2012.

[7] S. Waldert, "Invasive vs. non-invasive neuronal signals for brain-machine interfaces: Will one prevail?" *Frontiers in Neuroscience*, vol. 10, p. 295, 2016. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnins.2016.00295

[8] B. Pesaran, M. Vinck, G. T. Einevoli, A. Sirota, P. Fries, M. Siegel, W. Truccolo, C. E. Schroeder, and R. Srinivasan, "Investigating large-scale brain dynamics using field potential recording: analysis and interpretation." *Nature Neuroscience*, vol. 21, no. 8, pp. 903–919, Jul. 2018.

[9] G. T. Einevoll, C. Kayser, N. K. Logothetis, and S. Panzeri, "Modelling and analysis of local field potentials for studying the function of cortical circuits." *Nature Neuroscience*, vol. 14, no. 8, pp. 770–785, Nov. 2013.

[10] S. D. Stavisky, J. C. Kao, P. Nuyujukian, S. I. Ryu, and K. V. Shenoy, "A high performing brain–machine interface driven by low-frequency local field potentials alone and together with spikes," *Journal of Neural Engineering*, vol. 12, no. 3, p. 036009, Jun. 2015.

[11] H. Lindén, T. Tetzlaff, T. C. Potjans, K. H. Pettersen, S. Grün, M. Diesmann, and G. T. Einevoll, "Modeling the spatial reach of the LFP." *Neuron*, vol. 72, no. 5, pp. 859–872, Dec. 2011.

[12] D. A. Markowitz, Y. T. Wong, C. M. Gray, and B. Pesaran, "Optimizing the decoding of movement goals from local field potentials in macaque cortex," *Journal of Neuroscience*, vol. 31, no. 50, pp. 18 412–18 422, 2011.

[13] A. Jackson and T. M. Hall, "Decoding Local Field Potentials for Neural Interfaces," *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 25, no. 10, pp. 1705–1714, 2017.

[14] M. Angjelichinoski, T. Banerjee, J. Choi, B. Pesaran, and V. Tarokh, "Minimax-optimal decoding of movement goals from local field potentials using complex spectral features," *Journal of Neural Engineering*, vol. 16, no. 4, May 2019.

[15] T. Banerjee, J. Choi, B. Pesaran, D. Ba, and V. Tarokh, "Classification of local field potentials using gaussian sequence model," in *2018 IEEE Statistical Signal Processing Workshop (SSP)*, June 2018, pp. 683–687.

[16] I. M. Johnstone, *Gaussian estimation : Sequence and wavelet models*, 2017.

[17] A. B. Tsybakov, *Introduction to Nonparametric Estimation*, 1st ed. Springer Publishing Company, Incorporated, 2008.

[18] H. Ramoser, J. Muller-Gerking, and G. Pfurtscheller, "Optimal spatial filtering of single trial eeg during imagined hand movement," *IEEE Transactions on Rehabilitation Engineering*, vol. 8, no. 4, pp. 441–446, Dec 2000.

[19] J. Farquhar, "A linear feature space for simultaneous learning of spatio-spectral filters in bci," *Neural Networks*, vol. 22, no. 9, pp. 1278 – 1285, 2009, brain-Machine Interface. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S089360800900152X

[20] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten, "Multiclass brain–computer interface classification by riemannian geometry," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 4, pp. 920–928, April 2012.

[21] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten, "Classification of covariance matrices using a riemannian-based kernel for bci applications," *Neurocomputing*, vol. 112, pp. 172 – 178, 2013, advances in artificial neural networks, machine learning, and computational intelligence. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231213001574

[22] F. Yger, M. Berar, and F. Lotte, "Riemannian approaches in brain-computer interfaces: A review," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 10, pp. 1753–1762, Oct 2017.

[23] Z. Zhang, G. Dai, C. Xu, and M. I. Jordan, "Regularized discriminant analysis, ridge regression and beyond," *J. Mach. Learn. Res.*, vol. 11, pp. 2199–2228, Aug. 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1756006.1859927

[24] J. H. Friedman, "Regularized discriminant analysis," *Journal of the American Statistical Association*, vol. 84, no. 405, pp. 165–175, 1989. [Online]. Available: http://www.jstor.org/stable/2289860

[25] J. Zhang, C. Yan, and X. Gong, "Deep convolutional neural network for decoding motor imagery based brain computer interface," in *2017 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, Oct 2017, pp. 1–5.

[26] H. Yang, S. Sakhavi, K. K. Ang, and C. Guan, "On the use of convolutional neural networks and augmented csp features

for multi-class motor imagery of eeg signals classification," in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Aug 2015, pp. 2620–2623.

[27] D. Sussillo, P. Nuyujukian, J. M. Fan, J. C. Kao, S. D. Stavisky, S. Ryu, and K. Shenoy, "A recurrent neural network for closed-loop intracortical brain–machine interface decoders," *Journal of Neural Engineering*, vol. 9, no. 2, p. 026027, mar 2012.

[28] D. Sussillo, S. D. Stavisky, J. C. Kao, S. I. Ryu, and K. V. Shenoy, "Making brain–machine interfaces robust to future neural variability," *Nature communications*, vol. 7, p. 13749, 2016.

[29] C. Pandarinath, D. J. O'Shea, J. Collins, R. Jozefowicz, S. D. Stavisky, J. C. Kao, E. M. Trautmann, M. T. Kaufman, S. I. Ryu, L. R. Hochberg, J. M. Henderson, K. V. Shenoy, L. F. Abbott, and D. Sussillo, "Inferring single-trial neural population dynamics using sequential auto-encoders," *bioRxiv*, 2017. [Online]. Available: https://www.biorxiv.org/content/early/2017/06/20/152884

[30] N. Tiwari, D. R. Edla, S. Dodia, and A. Bablani, "Brain computer interface: A comprehensive survey," *Biologically Inspired Cognitive Architectures*, vol. 26, pp. 118 – 129, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2212683X18301142

[31] K. HALTAŞ, A. ERGÜZEN, and E. ERDAL, "Classification methods in eeg based motor imagery bci systems," in *2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, Oct 2019, pp. 1–5.

[32] S. Aggarwal and N. Chugh, "Signal processing techniques for motor imagery brain computer interface: A review," *Array*, vol. 1-2, p. 100003, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2590005619300037

[33] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. The MIT Press, 2016.

[34] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006.

[35] F. Lotte, "Signal processing approaches to minimize or suppress calibration time in oscillatory activity-based brain–computer interfaces," *Proceedings of the IEEE*, vol. 103, no. 6, pp. 871–890, June 2015.

[36] V. Jayaram, M. Alamgir, Y. Altun, B. Scholkopf, and M. Grosse-Wentrup, "Transfer learning in brain-computer interfaces," *IEEE Computational Intelligence Magazine*, vol. 11, no. 1, pp. 20–31, Feb 2016.

[37] P. Welch, "The use of fast fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms," *IEEE Transactions on Audio and Electroacoustics*, vol. 15, no. 2, pp. 70–73, June 1967.

[38] F. J. Massey Jr., "The kolmogorov-smirnov test for goodness of fit," *Journal of the American Statistical Association*, vol. 46, no. 253, pp. 68–78, 1951. [Online]. Available: https://amstat.tandfonline.com/doi/abs/10.1080/01621459.1951.10500769

[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014.