Deep Learning Neural Encoders for Motor Cortex

Ken-Fu Liang, *Student Member, IEEE*, and Jonathan C. Kao, *Member, IEEE*

***Abstract*—Intracortical brain-machine interfaces (BMIs) transform neural activity into control signals to drive a prosthesis or communication device, such as a robotic arm or computer cursor. To be clinically viable, BMI decoders must achieve high accuracy and robustness. Optimizing these decoders is expensive, traditionally requiring animal or human experiments spanning months to years. This is because BMIs are closed-loop systems, where the user updates his or her motor commands in response to an imperfectly decoded output. Decoder optimization using previously collected "offline" data will therefore not capture this closed-loop response. An alternative approach to significantly accelerate decoder optimization is to use a closed-loop experimental simulator. A key component of this simulator is the neural encoder, which synthetically generates neural population activity from kinematics. Prior neural encoders do not model important features of neural population activity. To overcome these limitations, we use deep learning neural encoders. We find these models significantly outperform prior neural encoders in reproducing peri-stimulus time histograms (PSTHs) and neural population dynamics. We also find that deep learning neural encoders better match neural decoding results in offline data and closed-loop experimental data. We anticipate these deep-learning neural encoders will substantially improve simulators for BMIs, enabling faster evaluation, optimization, and characterization of BMI decoder algorithms.**

***Index Terms*—Brain-machine interfaces, neural encoders, neural networks, motor cortex, deep learning.**

I. Introduction

GENERATING neural activity has wide implications in systems neuroscience and neural engineering. Neural encoding models (encoders) have produced insight into computations in early visual processing stages (e.g., [1]–[4]). Recent work has applied deep learning, specifically convolutional neural networks (CNNs), to model retinal activity. Analysis of these CNNs revealed roles for feedforward inhibition, recurrent lateral connections, and noise in explaining empirical neural responses [5]–[7]. Neural encoders have been used to argue that the motor cortex does not represent movement, but is rather a dynamical system (e.g., [8], [9]). In engineering applications, neural encoders are proposed to augment memory function through modeling hippocampal activity [10] and deliver realistic sensory sensation through stimulation of sensory cortices [11]–[13]. Finally, motor cortical neural encoders are a critical component for simulating motor brain-machine interfaces (BMIs) [14].

We focus on building motor cortical neural encoders for intracortical motor BMI simulation, with the goal of substantially accelerating clinical translation. For people with motor neurological disease and injury, such as amyotrophic lateral sclerosis (ALS) and spinal cord injury [15]–[17], BMIs aim to recover lost motor function and communication [18]–[22]. The ultimate goal is to improve quality of life for people with paralysis by providing direct neural control of prosthetic arms or computer cursors. These prostheses are guided by a control signal decoded from motor regions of the brain (e.g., [21], [23], [24]) as illustrated in Fig. 1(a). In clinical trials, these neural decoders are trained with recorded neural signals during imagined (or intended) movements [20]. While considerable research over the past 15 years has led to important BMI demonstrations [19]–[22], [25]–[31] several challenges remain towards achieving clinically viable BMI systems.

To be clinically viable, intracortical BMIs must achieve high performance at a level justifying neurosurgery [32]. The performance of these systems is heavily impacted by the BMI decoder. Recent efforts have produced novel algorithms to increase BMI performance and robustness by utilizing feedback control [33], [34], neural population dynamics [35]–[37], latent factors [38]–[44], and deep learning [45]. Yet, BMI development remains relatively slow. An important reason is that the most reliable test of BMI decoder performance are closed-loop experiments, which are expensive and time-consuming. In particular, BMIs are unique in that a user interacts with the system, adjusting his/her motor commands in response to observing the BMI decoder output. When the decoded output does not match user intent, the user updates his/her motor commands. This may cause the statistics of the neural activity to be different to previously collected open-loop experimental data ("offline evaluation") used to train the decoder [46]–[49]. It is therefore possible for decoders optimized through offline evaluation to be suboptimal in closed-loop experiments [14], [50], [51].

Yet, closed-loop experiment also face challenges. They have greater costs per tested algorithm than offline decoding, being

Manuscript received July 31, 2019; revised October 3, 2019; accepted November 13, 2019. Date of publication November 25, 2019; date of current version July 17, 2020. This work was supported by a UCLA Hellman Fellowship (J.C.K.), a UCLA Dean's Scholarship and Living Spring Fellowship (K.L.), and a UCLA Computational Medicine Amazon Web Services grant. *(Corresponding author: Jonathan C. Kao.)*

K.-F. Liang is with the Department of Electrical and Computer Engineering, University of California-Los Angeles.

J. C. Kao is with the Department of Electrical and Computer Engineering, and the Neuroscience Program, University of California-Los Angeles, Los Angeles, CA 90095 USA (e-mail: kao@seas.ucla.edu).

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/TBME.2019.2955722

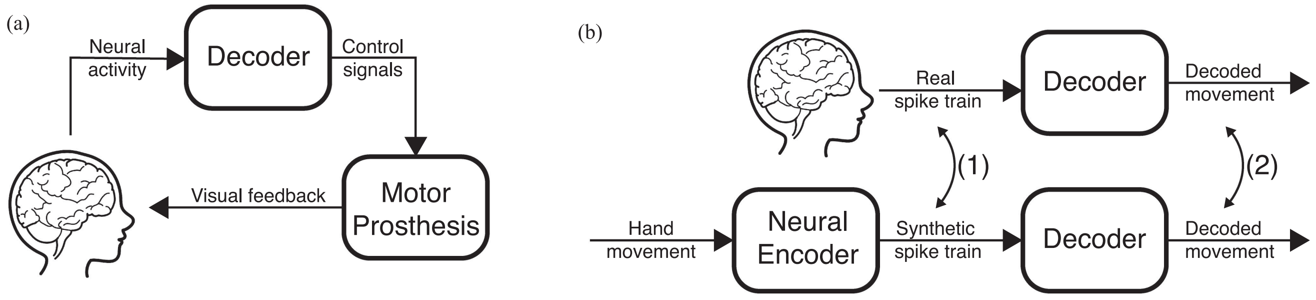

Fig. 1. BMI system and neural encoder. (a) Neural activity is recorded from electrode arrays implanted in motor cortex. A decoder transforms neural activity into control signals which guides a motor prosthesis. The BMI user receives visual and potentially sensory feedback, adjusting his or her control policy in response to observing the decoded movement. (b) A neural encoder takes user hand movement kinematics and produces synthetic neural activity. A good encoder reproduces single electrode activity, neural population dynamics, and decoded movement decoded from real neural activity. Here, we compare (1) the PSTHs and neural dynamics of recorded and synthetic neural activity, and (2) decoded movements from real and synthetic neural activity.

more time-consuming and financially expensive [14]. Further, closed-loop experiments incorporate sources of variation including animal behavior and motivation. While this variance can be beneficially used to increase decoder robustness [45], this variation may also increase the time and data required to demonstrate statistically significant differences. Finally, closed-loop experiments are difficult to compare across laboratories due to variations in recording technology, animal training, and task design. These obstacles are largely not faced by offline decoding and evaluation, where decoders are trained and tested on previously recorded motor cortical data. Offline evaluation, in contrast, allows for the same dataset to be used for algorithm development, has reduced variability from animal motivation, and can be performed at time scales that are orders of magnitude smaller than closed-loop experiments. Offline evaluation can therefore play an important role as stepping stones to closed-loop experiments, producing insight into what algorithms are most promising to evaluate in available closed-loop experiments. However, it is important to emphasize that offline decoding does not capture the closed-loop nature of BMI systems [14], [50]–[53].

In contrast to closed-loop animal experiments and offline decoding, prior work introduced a middle ground: neural activity can be simulated to more rapidly perform experiments that incorporate the closed-loop nature of BMI systems. This online prosthetic simulator (OPS) [14] enables faster decoder evaluation incorporating closed-loop feedback control policies without requiring intracortical recordings from the brain. Cunningham and colleagues showed using OPS that BMI performance should increase with smaller bin width, an empirical result they and another study confirmed [34]. In the OPS, spike trains were synthesized via an inhomogeneous Poisson process whose rate was described by a speed-modulated tuning curve model [54]. We call this model the Poisson Process Velocity Tuning (PPVT) model. Several studies have demonstrated that this model does not reproduce complex heterogeneity in neuron firing rates, including changing preferred directions with time [55], nonlinear structure in the neural population activity [56], and neural population dynamics (e.g., [9], [57]).

In this work, we introduce deep learning (DL) based neural encoders for synthesizing motor cortical neural population activity from kinematic behavior. We chose to use DL due to its success in a variety of research areas including, but not limited to, computer vision [58]–[60], object detection [61]–[64], neural signal denoising [56], early visual representations modeling [65], and sensory cortex modeling [7]. We trained DL encoders in a supervised fashion using data collected while a monkey performed a reaching task. We demonstrate that DL encoders can better reproduce single neuron PSTH variability, neural population dynamics, and neurally decoded movements. Further, we show that these models more faithfully reproduce closed-loop BMI kinematics decoded from a monkey experiment. We anticipate that DL encoders will be a key component in developing BMI simulators. Together, with offline decoding experiments, closed-loop simulation may substantially accelerate the evaluation and optimization of BMIs.

II. Methods

A. Experimental Setup

All surgical and animal care procedures were performed in accordance with National Institutes of Health guidelines and were approved by the Stanford University Institutional Animal Care and Use Committee. Experiments were conducted with two adult male rhesus macaques (Monkeys J and L). Monkey J (L) was implanted with two (one) 96 electrode Utah arrays (Blackrock Microsystems Inc., Salt Lake City, UT) using standard neurosurgical techniques. Monkey J's arrays were implanted in dorsal premotor cortex (PMd) and primary motor cortex (M1) as visually estimated from local anatomical landmarks, while Monkey L's array was implanted around the PMd/M1 border. Monkey J's (L's) arrays were implanted 75 (94) months prior to data collection for this work. The monkeys made point-to-point reaches in a 2D plane with a virtual cursor controlled by the contralateral arm or by a BMI. The experimental setup has been previously described (e.g., [14], [66]). The virtual cursor and targets were presented in a three-dimensional (3D) environment (MusculoSkeletal Modeling Software (MSMS), Medical Device Development Facility (MDDF), USC, Los Angeles, CA). Hand position data were measured with an infrared reflective bead tracking system (Polaris, Northern Digital, Ontario, Canada). Spike counts were collected by applying a single threshold, set to $-4.5\times$ the root-mean-square of the high-pass filtered spike voltage per electrode [67]. Behavioral control and neural decode were run on separate PCs using Simulink/xPC platform (Mathworks, Natick, MA) with communication latencies of 3 ms. This enabled millisecond timing precision for all computations. Neural data were initially processed by the Cerebus recording system (Blackrock Microsystems Inc., Salt Lake City, UT) and were available to the behavioural control system within 5 ± 1 ms. Visual presentation was provided via two LCD monitors with refresh rates at 120 Hz, yielding frame updates of 7 ± 4 ms. Two mirrors visually fused the displays into a single 3D percept for the user, creating a Wheatstone stereograph. All tasks presented in this study were restricted to a two-dimensional plane. We report results from Monkey J in the main manuscript. All analyses described were reproduced and confirmed in Monkey L, which is shown in the supplementary material.

B. Tasks

We used recordings from when monkeys performed a center-out-and-back reaching task. In this task, eight targets were placed with uniform spacing on the circumference of a 12-cm radius circle. The monkey acquired the center target followed by one of the eight (randomly chosen) radial targets. Each target had to be acquired within 2 seconds, or the trial was counted as a failure. After successful acquisition of a radial target, or following the failure to acquire any radial target, the center target was prompted. Each target had a 4-by-4 cm acceptance window centered around the target. For every target selection, the subject had to hold the cursor within the acceptance window for 500 contiguous milliseconds.

In open-loop experiments, the cursor position was the monkey's hand position. In closed-loop (BMI) experiments, the cursor's movements were decoded by a Recalibrated Feedback Intention-Trained Kalman Filter (ReFIT-KF) decoder [33] with neural spikes summed in 10 ms bins. The monkey's arm was free to move during BMI experiments [49], [68]. The ReFIT-KF decoder uses kinematic intention estimation and closed-loop adaptation to better estimate decoded velocity [46]. In open- and closed-loop experiments, we recorded monkey's hand positions and neural signals for training neural encoders. Open-loop data and closed-loop data were recorded on the same day to minimize neural variation and allow a direct comparison of closed-loop decoding from real data versus simulated data. In total, we used approximately 500 successful trials from open-loop experiments and 150 successful trials from closed-loop experiments to train and test neural encoders.

C. Training and Testing Data

When evaluating neural encoders on open-loop data, the training set comprised approximately 80% of open-loop trials; the remaining open-loop trials comprised the testing set. Trials were randomly chosen to be training or testing set trials. We repeated experiments with different training and testing data splits. No conclusions of the paper were affected based on how the training and testing data were split. We also performed experiments evaluating neural encoders on closed-loop data. In these experiments,the training set comprised the open-loop data and the test set comprised the closed-loop data.

D. Comparison Metrics

1) Pearson Correlation Coefficient (PCC) of PSTH: We report the PCC between electrode and synthetic PSTHs. The PSTH is the average firing rate for reaches to each of eight reach conditions. This was calculated by averaging spike counts across all trials within a condition, aligned to target onset. We concatenated the PSTH for each of eight reach conditions into a vector. The data were binned at intervals of 25 ms; each reach condition comprised 32 bins, or 800 ms of activity. Hence, each vector comprised 256 data points. We then calculated the PCC between these vectors for real and synthetic PSTHs.

2) Mean Squared Error (MSE) of Neural Trajectories: We report the MSE of neural trajectories between real and synthetic neural population activity. The neural trajectory provides a low-dimensional and compact representation of the high-dimensional recorded activity through time [69]–[72]. We first calculated the principal components (PCs) of real neural population activity. Subsequently, we projected both real and synthetic neural population into the top 7 PCs, which captured at least 90% of the variance. We then calculated the MSE between these trajectories.

3) Normalized Mean Squared Error of Decoded Position: We computed MSE between positions decoded from the real and synthetic neural activity. We subsequently normalized the MSE (NMSE) by dividing by the variance of decoded position from real neural data. We evaluated NMSE for common decoders in the literature, including the optimal linear estimator (OLE) (e.g., [36], [73]), Wiener filter (WF) (e.g., [20], [36], [74]), and Kalman filter (KF) (e.g., [22], [33], [66], [75], [76]). Briefly, the OLE is the least-squares mapping from neural data to kinematics; the WF is the least-squares mapping from a history of neural activity to kinematics; and the KF models a linear dynamical system where the states are kinematics and the observations are neural activity.

E. Dimensionality Reduction

1) Principal Component Analysis (PCA): We used PCA to reduce the dimensionality of data and compare neural trajectories. PCA is an orthogonal transformation of the neural data that maximizes the variability of the data in low-dimensional projections. We performed PCA on neuron PSTHs to emphasize across-condition variability over single-trial variability. When comparing neural trajectories, we projected both real and synthetic neural population activity into the PCs found from real data.

2) jPCA: jPCA, a rotation of the top PCs, finds an orthonormal basis that reveals rotational structure in data [8]. We applied jPCA after finding the top six PCs, since we found that 7 PCs capture over 90% of the PSTH variance. This comprised finding a skew symmetric matrix least-squares mapping from the position of the neural trajectory to its velocity. We report R^2_{skew}, which is the variance explained in predicting the neural trajectory velocity from its position. A higher R^2_{skew} indicates that the system is better described by rotational dynamics. We also report $R^2_{\text{skew}}/R^2_{\text{best}}$, where R^2_{best} is the variance explained using unconstrained least-squares [57]. This indicates how well rotational dynamics describe data relative to unconstrained linear dynamics.

F. Encoding Models

Neural encoders transform kinematics, x_t, into binned spike counts, y_t. All presented models generate a neural firing rate. Neural encoders were trained to reproduce the empirical binned spike counts recorded from multi-unit threshold crossing activity. We did not reproduce single unit activity since decoding threshold crossings achieves similar performance compared to decoding after spike sorting [77], [78] and may be more robust [67]. Further, BMI clinical trials largely decode multiunit activity [18], [21], [22], [66], [75]. We calculated binned spike counts by treating the neural encoder firing rate as the rate of an inhomogeneous Poisson process. We evaluated several linear tuning models, including a preferred direction (PD) model, a Poisson process velocity tuning (PPVT) model, and generalized linear models (GLMs). We subsequently evaluated DL neural encoders, including a multilayer perceptron (MLP), a MLP with historical inputs (MLPh), and a recurrent neural network (RNN).

Below, unless otherwise stated, all DL models were trained in a supervised manner with recorded kinematics comprising 2D position and velocity (inputs) and neural activity (outputs) during the center-out-and-back task. Unless otherwise stated, kinematics and binned spike counts were evaluated at 25 ms bin width resolution. Networks were trained to optimize a maximum-likelihood cost function, described further below. This loss function was optimized with stochastic gradient descent, using the Adam optimizer [79]. All DL models were initialized with the Xavier uniform initializer [80]. We did not otherwise constrain the models; in particular there were no constraints on reproducing neural population dynamics.

1) Preferred Direction (PD) Model: The PD model is based on a tuning curve model where each neuron's firing rate is explained as a function of the reach angle, with the angle eliciting the highest firing rate called the "preferred direction" (PD) [54]. The PD model calculates firing rate based on a cosine tuning curve. Subsequently, binned spike counts, y_t, are generated by treating the PD model firing rate as the underlying rate of an inhomogeneous Poisson distribution. The equations to generate binned spike counts are:

$$\lambda_t = \lambda_o + (\lambda_{\max} - \lambda_o)\cos(\theta_t - \theta_{\max}) \tag{1}$$

$$\hat{y}_t|\lambda_t \sim \text{Poisson}\left(\max(0, \lambda_t)\right), \tag{2}$$

where: λ_t is the neural firing rate, y_t is the binned spike counts, θ_t is the reach angle, $\theta_{\max}$ is the PD, and λ_o is an offset firing rate. The neuron's modeled firing rate, λ_t, ranges from $2\lambda_o - \lambda_{\max}$ (when the reach angle is opposite to the PD) to $\lambda_{\max}$ (reach angle aligned to the PD). The model parameters were found using the technique of [54], where firing rates were averaged from 200 ms to 500 ms after trial initiation. Because neural firing rates cannot be negative, we draw spike counts with rates lower bounded by 0, i.e., with rate $\max(0, \lambda_t)$.

2) Poisson Process Velocity Tuning (PPVT) Model: The PPVT model extends the PD model by incorporating reach speed into the neural encoder. This was the model used by Cunningham and colleagues [14]. Here, the firing rate is linearly scaled based on the speed of the reach. Firing rate is calculated as,

$$\lambda_t = \lambda_o + (\lambda_{\max} - \lambda_o)\cos(\theta_t - \theta_{\max}) \cdot s_t. \tag{3}$$

In this equation, s_t is the scaled movement speed at time t. We scale s_t so that the firing rates, when decoded, produce reasonable trajectories. This scaling is subject dependent, since different subjects may reach with different vigor. We generate binned spike counts using equation (2).

3) Generalized Linear Models (GLMs): The GLM model is a flexible generalization of ordinary linear regression [81]–[83]. In its most basic form, we calculate the rates as:

$$\lambda_t = k \cdot x_t + \text{noise}, \tag{4}$$

where k is a vector of weights and x_t are the kinematic inputs. Unless otherwise stated, x_t is the 2D position and velocity of the hand at time t. The GLM can be extended to incorporate different noise distributions and a link function relating inputs to the rates. In total, we evaluated: (1) Linear Gaussian GLM, with Gaussian noise and identity link function; (2) Linear Gaussian GLM with 4 bins of kinematic input history, so that the input was $(x_t, x_{t-1}, \ldots, x_{t-4})$ (GLMh), (3) a Poisson GLM with a log link function, and (4) a Binomial GLM with a logistic link function. We performed maximum likelihood estimation through iteratively reweighted least squares. In the main manuscript, GLM and GLMh refer to linear Gaussian GLMs.

4) Multilayer Perceptron (MLP) Model: The multilayer perceptron (MLP) model is a nonlinear, fully connected feed-forward neural network, as shown in Fig. 2(a). As a feedforward network, it does not model any firing rate dynamics but enables a data-driven, nonlinear approach to predict firing rates from kinematics. The MLP is a universal function approximator and has higher capacity than the linear neural encoders [84]. The MLP model takes kinematics, x_t, and generates firing rates, λ_t. We denote an MLP layer as

$$h_t = \text{MLP}^f_N(x_t) \tag{5}$$

$$= f(Wx_t + b), \tag{6}$$

where f is activation function, N is the number of neurons, $W \in \mathbb{R}^{N\times \dim(x_t)}$, $b \in \mathbb{R}^N$, and $h_t \in \mathbb{R}^N$. We used the sigmoid activation function, i.e., $f(x) = \sigma(x) = \frac{1}{1+\exp(-x)}$ in all hidden layers. Because firing rates are non-negative and not bounded by 1, we used an exponential nonlinear activation function in the final layer, i.e., $f(x) = \exp(x)$. The overall neural encoder we tested is:

$$\lambda_t = \text{MLP}^{\exp}_{192}(\text{MLP}^{\sigma}_{128}(\text{MLP}^{\sigma}_{64}(\text{MLP}^{\sigma}_{32}(x_t)))), \tag{7}$$

with spikes generated according to equation (2). To optimize the model, we maximized the log-likelihood of the empirical data by assuming that empirical binned spike counts, y_t, were Poisson distributed and conditionally independent given the firing rate, λ_t. Optimization was performed in batches of trials. The

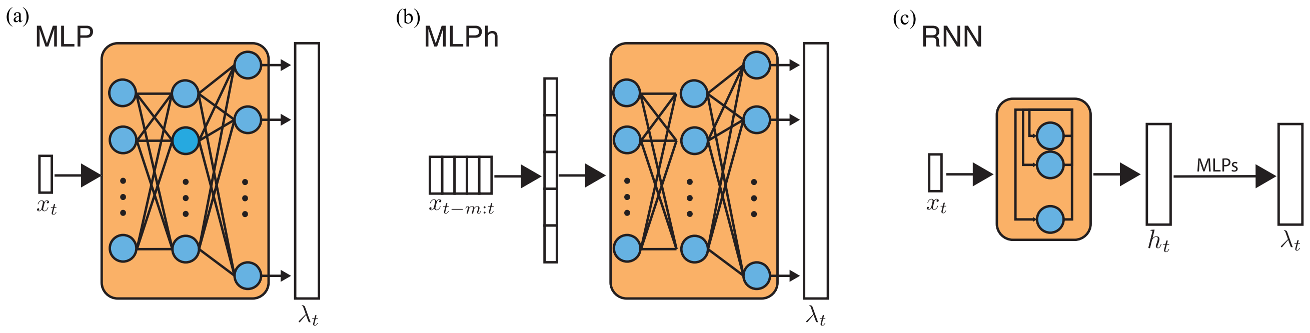

Fig. 2. (a) Multilayer perceptron (MLP) architecture. The MLP consists of an input layer, three hidden layers, and an output layer. The nodes are fully connected. (b) MLP with history (MLPh). The MLPh processes a history of inputs with a larger input layer but otherwise has the same architecture as the MLP. (c) Recurrent neural network (RNN) model. The RNN processes an input history, but incorporates recurrent dynamics. Its output is passed to MLP layers.

log-likelihood function is,

$$\mathcal{L} = \sum_i \sum_t \left(y_t^i \log \lambda_t - \lambda_t \right), \tag{8}$$

where i is iterating over trials in a batch, t is iterating over the time in trial i, and y_t^i is the binned spike counts at time t on trial i.

5) MLP With Kinematics History (MLPh) Model: While the MLP is a nonlinear approach, it does not incorporate any historical kinematics into its predictions. Without kinematic history, the MLP generates neural activity from only positions and velocities at that time. For example, the MLP will produce the exact same neural activity with the exact same input at time t, irrespective of past data. Neural population activity from motor cortex is dynamical, having structure over time. To account for temporal structure, we tested an MLP model inputting a history of kinematics. We call this encoder MLPh, with the h denoting historical inputs. The MLPh model inputs the vertical concatenation of the kinematics, $(x_t, x_{t-1}, \ldots, x_{t-m})$, and outputs firing rate, λ_t. We chose $m = 4$ through heuristic optimization over several values of m. We used the same activation function structure as in the MLP model. Hence, firing rates were generated via the following equation:

$$\lambda_t = \text{MLP}_{192}^{\exp}(\text{MLP}_{128}^{\sigma}(\text{MLP}_{64}^{\sigma}(\text{MLP}_{32}^{\sigma}([x_{t-m:t}])))), \tag{9}$$

with spikes generated according to equation (2). Like in the MLP, we performed optimization by maximizing the log-likelihood of the observed binned spike counts. The MLPh is illustrated in Fig. 2(b).

6) RNN Model: The MLPh enables temporal structure by processing a history of inputs, but does not explicitly incorporate a dynamical model. To model dynamics, we trained an RNN model to process the inputs, x_t, and generate firing rates, λ_t. The RNN incorporates history by maintaining a hidden state, h_t. We use RNN_N^f to denote the RNN architecture, Concretely,

$$z_t = RNN_N^f(x_t) \tag{10}$$
$$= W_o h_t + b_o \tag{11}$$
$$h_t = f(W_f x_t + W_r h_{t-1} + b_h), \tag{12}$$

where $W_f \in \mathbb{R}^{N \times \dim(x_t)}$ maps the inputs to the hidden state, $h_t \in \mathbb{R}^N$, $W_r \in \mathbb{R}^{N \times N}$ captures the RNN's recurrent dynamics, $W_o \in \mathbb{R}^{N \times N}$ linearly reads out the hidden state, $z_t \in \mathbb{R}^N$ is the output of the network, and $b_h \in \mathbb{R}^N$ and $b_o \in \mathbb{R}^N$ are learned biases. we ran the RNN for 32 time-steps to seed the hidden state, h_t, then evaluated the log-likelihood of the model for training. In experiments, we found that additional nonlinearity at the output increased performance. In the reported results, we therefore added MLP layers to the RNN output. The complete RNN neural encoder is:

$$\lambda_t = \text{MLP}_{192}^{\exp}(\text{MLP}_{192}^{\sigma}(\text{MLP}_{192}^{\tanh}(\text{RNN}_{192}^{\tanh}(x_t)))) \tag{13}$$

with spikes generated according to equation (2). The RNN was trained by maximizing the log-likelihood of the observed binned spike counts under the assumption that they follow Poisson statistics. The RNN architecture as illustrated in Fig. 2(c).

III. Results

We found that deep learning (DL) models outperformed representational tuning models (PD, PPVT) and GLM models (GLM, GLMh) in reproducing recorded single electrode activity, neural population activity, and decoded kinematics. Importantly, we decoded closed-loop data to demonstrate generalization of neural encoders to closed-loop BMI data it was not trained on. Here, we detail these analyses and comparisons.

A. DL Encoders Better Reproduce PSTHs

We first evaluated how each encoding model reproduced every electrode's PSTHs. We also calculated PSTHs for synthetic spike trains for each encoding model. We subsequently evaluated the similarity between recorded and synthetic electrode PSTHs by calculating the PCC (also denoted Pearson's r, see Methods). A model's overall Pearson's r is the average of the Pearson's r across all 192 (96) electrodes for Monkey J (L).

Fig. 3 illustrates a representative example, with the recorded PSTHs shown in panel (**h**) and neural encoder PSTHs shown in other panels. As has been previously described, the PSTHs have a condition-independent increase in activity followed by heterogeneous activity that may be multiphasic [85]. The PD and PPVT models fail to reproduce these characteristics. We found

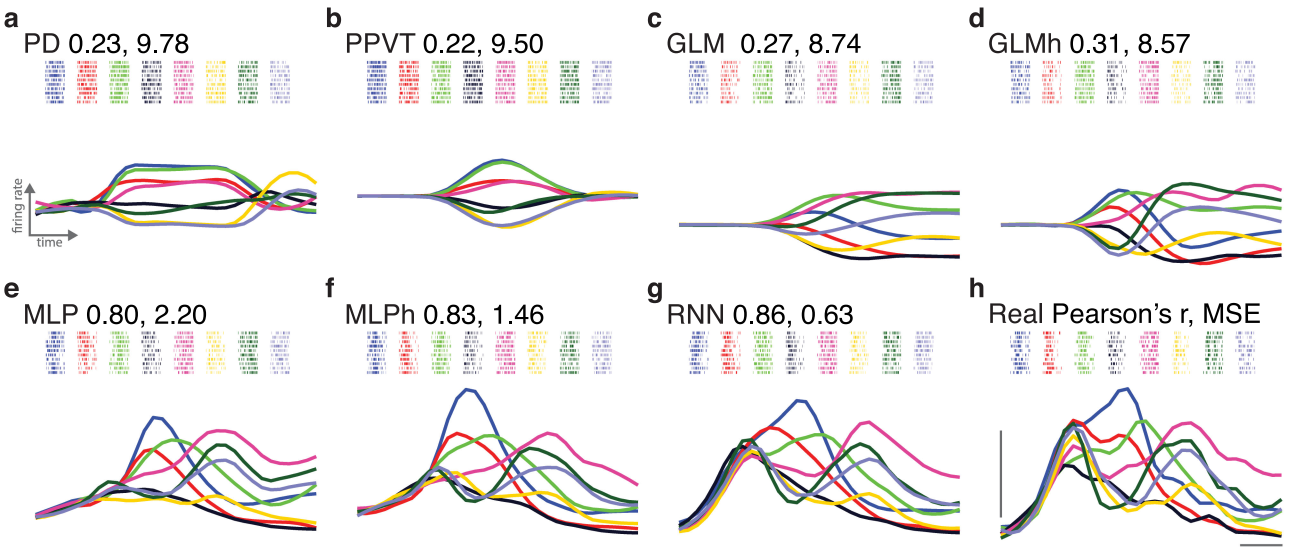

Fig. 3. PSTHs and spike rasters of recorded and synthetic neural activity. The PSTHs are of: (a) PD, (b) PPVT, (c) GLM, (d) GLMh, (e) MLP. (f) MLPh, (g) RNN, (h) a randomly chosen electrode from empirical recordings. (a)-(g) show the corresponding neural encoder outputs in reproducing electrode activity in (h). Each color shows the average firing rate to one of eight center-out reach condition over time, with eight different colors corresponding to eight reach conditions. In (h), the vertical bars denote 50 spikes/s, and the horizontal bar denotes 100 ms. The average PCC across all neurons and average MSE of PSTH (spikes^2/s) for each encoding model are shown in each panel. We also show spike rasters in each panel. The PCC for single-trial binned spike counts for PD, PPVT, GLM, GLMh, MLP, MLPh and RNN were 0.016, 0.011, 0.020, 0.024, 0.103, 0.135, and 0.137, respectively. Their respective MSEs were 53.60, 52.99, 41.65, 41.51, 39.62, 38.90, and 40.77 spikes^2/s.

that the PD model achieved relatively static firing rates (Fig. 3(a), Pearson's $r = 0.23$, MSE $= 9.78$ spikes^2/s). This is expected because the PD model's firing rates only vary with reach angle. The PPVT incorporates speed modulation, causing the PSTHs to resemble the speed profile. However, consistent with prior work, the PPVT encoder does not capture heterogeneity, including multiphasic behavior, in PSTH activity [8], [9], [55]. The PPVT achieves comparable performance to PD (Fig. 3(b), Pearson's $r = 0.22$, MSE $= 9.50$ spikes^2/s, not significantly different from PD model, Wilcoxon rank-sum test over repeatedly trained models).

We next evaluated GLM models, which incorporate position and velocity. We found that these models achieved constant firing rates while approaching targets. (Fig. 3(c), Pearson's $r = 0.27$, MSE $= 8.74$ spikes^2/s, not significantly different from PPVT model). When evaluating GLMh, which incorporates history of kinematics, we saw multiphasic PSTHs (Fig. 3(d), Pearson's $r = 0.31$, MSE $= 8.57$ spikes^2/s).

In contrast to representational tuning models and GLM models, we found that DL models better reproduced empirically recorded PSTHs. The MLP was capable of reproducing a condition-independent increase and qualitative motifs in the original PSTH, including the relative ordering of condition firing rates. However, it does not capture all multiphasic activity, such as the biphasic activity of the green PSTH (Fig. 3(e), Pearson's $r = 0.80$, MSE $= 2.20$ spikes^2/s, better than representational models, both $p < 10^{-7}$, Wilcoxon rank-sum test). As discussed in the Methods, we reasoned that a key limitation of tuning and MLP models is that they do not incorporate history over the inputs, i.e., kinematics. (It is worth noting that PSTHs from the MLP model would be symmetric if only provided with symmetric velocity inputs, as is the case for center-out reaches; providing position information enables the MLP to capture richer and non-symmetric PSTHs.) We therefore evaluated the MLPh model, incorporating 4 time bins of kinematic history, and observed that MLPh better reproduced PSTHs (Fig. 3(f), Pearson's $r = 0.83$, MSE $= 1.46$ spikes^2/s, better than MLP model, both $p < 10^{-7}$, Wilcoxon rank-sum test).

Finally, a more general and effective method to model dynamics in neural population activity is through an RNN [9], [86], [87]. The RNN can generate different outputs for the same input depending on its internal hidden state, which is a function of past inputs and the RNN's own internal dynamics. We found that an RNN achieved better performance than the MLPh model (Fig. 3(g), Pearson's $r = 0.86$, MSE $= 0.63$ spikes^2/s, comparison to the MLPh model, both $p < 10^{-7}$, Wilcoxon rank-sum test). Together, these results demonstrate that DL models better reproduce single electrode firing rates.

B. DL Models Better Reproduce Neural Population Motifs

As DL models better reproduce empirical firing rates, we next wondered how well DL models reproduce neural population activity [8], [88]. It is worth noting that related work has demonstrated that RNNs trained to process task inputs (such as a go cue and target location) to produce either kinematics [9] or EMG [86], [87] exhibit low-dimensional and rotational dynamics, a motif also found in premotor and primary motor cortex [57], [89]. In our models, rather than evaluating the neural network's artificial activity, we evaluated neural population structure in the network output, i.e., firing rates, given kinematic inputs.

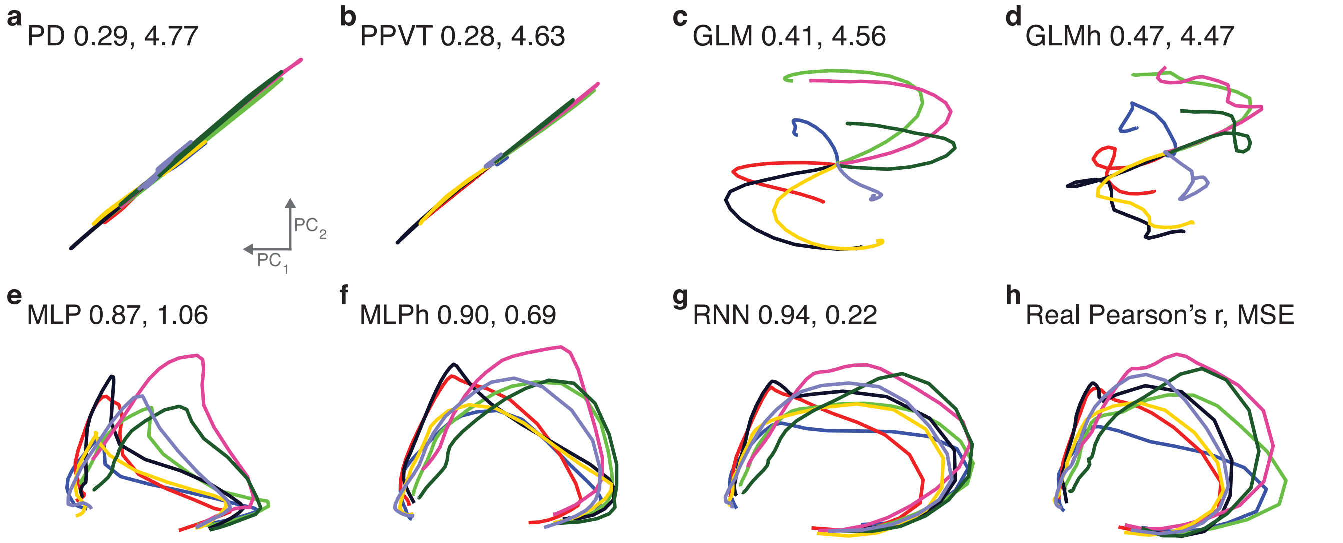

Fig. 4. Projections of PSTHs on PCs found from real neural data. The projections are of (a) PD, (b) PPVT, (c) GLM, (d) GLMh, (e) MLP, (f) MLPh, (g) RNN, (h) empirically recorded neural population activity. Each color shows the neural trajectory of one of eight center-out reach conditions over time. We calculate the MSE and PCC of neural trajectories in the top 7 PCs.

To quantify neural population activity, we performed PCA on empirically recorded and synthetically generated PSTHs (see Methods). When performing PCA on recorded PSTHs, we found 7 dimensions were required to capture over 90% of the recorded neural variance. Tuning models were overly simplistic, demonstrating a smaller dimensionality. When performing PCA on synthetic neural population activity generated by the PD and PPVT outputs, we found that only 2 PCs are required to capture nearly 100% variance. GLM requried 3, and GLMh required 4 dimensions to capture over 90% of the neural variance. On the other hand, we found that MLP, MLPh and RNN required 5 or 6 dimensions to capture over 90% of the neural variance, as shown in Fig 5. Together, these results demonstrate that DL models more closely match the dimensionality of real data, reflecting relatively greater variability in the neural population response compared to tuning and linear models.

How similar are neural population trajectories in these low-dimensions? We compared low-dimensional projections via PCA on the synthetic versus recorded neural population. We projected synthetic and neural activity onto the PCs found by performing PCA on the real data Fig. 4. We found that tuning models had very different projections in comparison to real data (compare Fig. 4(a),(b),(h)). Tuning model low-dimensional variance primarily resided on a 1-dimensional axis, reflecting a large degree of covariation. To quantify these results, we compared the mean PCC and MSE of PC trajectories in the top 7 dimensions (capturing greater than 90% of the variance) between synthetic and real neural activity. Tuning and GLM models had relatively modest PCCs less than 0.5. On the other hand, DL models more closely reproduced neural trajectories, achieving PCCs above 0.85. This demonstrates that DL models better reproduced empirical neural population motifs.

Another way to measure neural population structure is to compare rotational dynamics in the neural population via jPCA (e.g., [9], [14], [57], [89], see Methods). We applied jPCA and

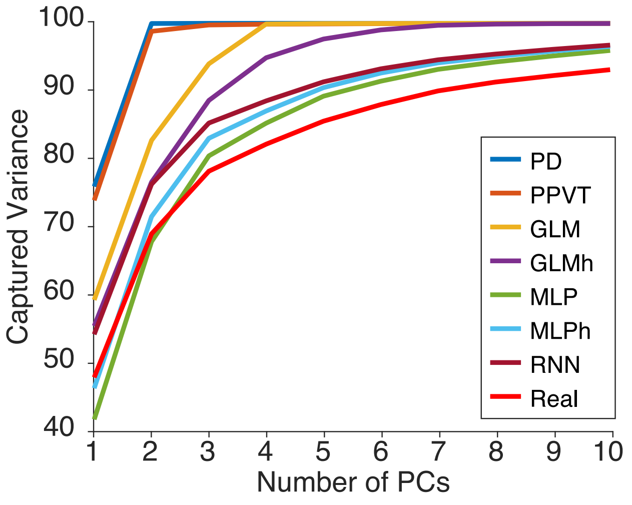

Fig. 5. Dimensionality of recorded and synthetic PSTHs. Only 2 PCs were needed to capture almost 100% variance of PD and PPVT activity. To capture over 90% of recorded neural variance, we required 3 PCs for GLM, 4 PCs for GLMh, and 5 or 6 PCs for DL models. The real data required 7 PCs.

found that neither the PD, PPVT or GLM activity could be well described by rotational dynamics ($R^2_{\text{skew}} < 0.01$ for PD, $R^2_{\text{skew}} < 0.01$ for PPVT, and $R^2_{\text{skew}} = 0.05$ for GLM; real data $R^2_{\text{skew}} = 0.35$, as shown in Fig. 6(a),(b),(c),(h)), as reported previously [9], [57]. We observed that GLMh showed stronger rotational dynamics ($R^2_{\text{skew}} = 0.28$), as shown in Fig. 6(d)). This is not entirely surprising, since models where neuron firing rate peaks occur at different times, with multiphasic activity, demonstrate rotational dynamics [9]. Interestingly, though the MLP model does not have internal dynamics or utilizes any historical data, we found that its generated PSTH exhibited a degree of rotational dynamics ($R^2_{\text{skew}} = 0.22$, Fig. 6(e)). This implies that, only considering the position and velocity at a time point t, a

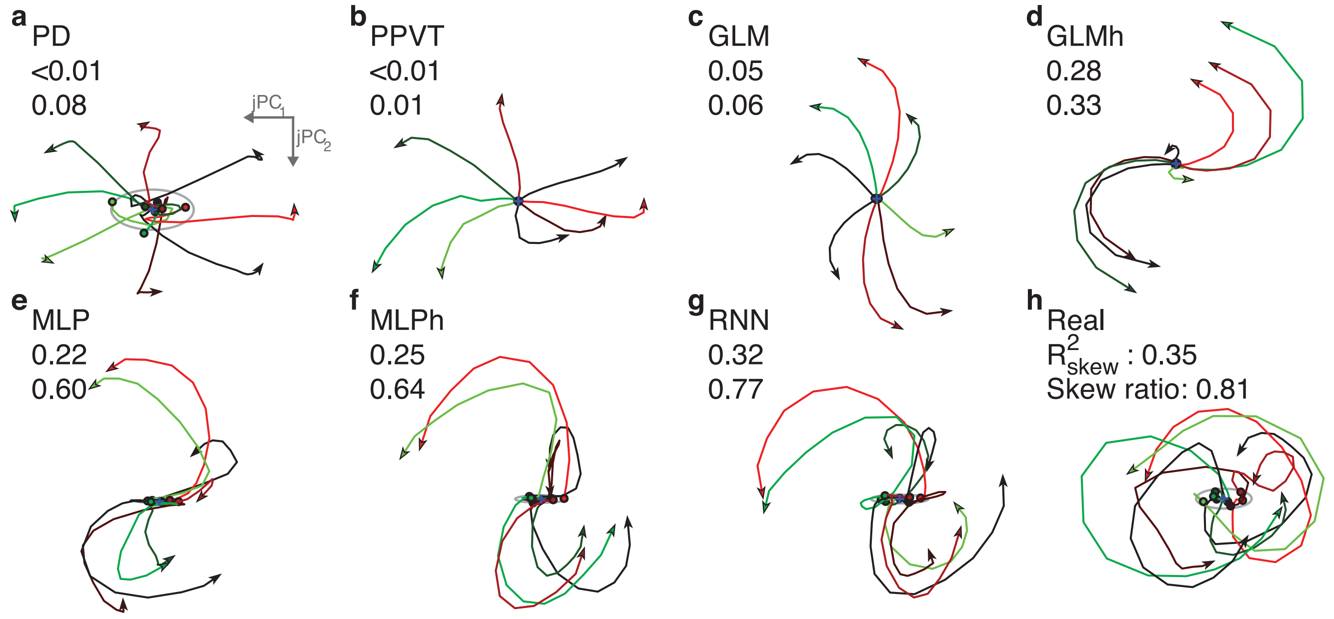

Fig. 6. jPCA projections of the population responses. Each plot shows the jPCA projection of eight center-out reach conditions (different colors). The jPCA projections are of: (a) PD, (b) PPVT, (c) GLM, (d) GLMh, (e) MLP, (f) MLPh, (g) RNN, (h) empirically recorded neural population activity. We show the R^2_{skew} in the jPCs as well as $R^2_{\text{skew}}/R^2_{\text{best}}$ (Skew ratio) for each encoding model (see Methods).

neural network can still generate neural population activity that exhibits rotational structure. We found that the degree of rotational dynamics in the encoding model output increased as the model was able to consider historical inputs; the MLPh exhibited more rotational dynamics than the MLP ($R^2_{\text{skew}} = 0.25$, Fig. 6(f), comparison to the MLP model, $p < 10^{-4}$, Wilcoxon rank-sum test). This was further improved by using RNN ($R^2_{\text{skew}} = 0.32$, Fig. 6(g), comparison to the MLPh model, $p < 10^{-7}$, Wilcoxon rank-sum test). These results demonstrate that DL models not only better reproduce single electrode PSTHs, but also better reproduce neural population motifs. Together, the data generated by DL models appears to be a more faithful representation of neural population activity in the motor cortex.

C. Kinematics Decoded From DL Generated Neural Activity Better Match Open-Loop Decoded Kinematics

As our motivating application for generating motor cortical neural signals is BMI simulation, we next assessed if DL synthetic neural data could better match offline decoding of previously recorded neural data. We therefore decoded neural data recorded from when monkeys performed a center-out-and-back reaching task, and compared these to kinematics decoded from encoding models. Neural activity from a more effective encoding model can be decoded to produce kinematics that more closely matched the kinematics decoded from real data. While DL encoding models more accurately reproduce PSTHs and population structure, as earlier described, optimal decoding dimensions may have little overlap with the top PCs of the activity [90]. Therefore, it is important to assess if neural variance in these kinematic dimensions is better captured by DL models. We trained an optimal linear estimator (OLE) [73], a Wiener filter (WF) [20], [74] and a Kalman filter (KF) [75], [76] to decode both recorded and synthetic neural activity. We show randomly chosen decoding trials in Fig. 7. The mean and standard deviation of the NMSE across all open-loop testing set trials is summarized in Fig. 8. We found that, across all decoders, DL generated neural activity more closely resembled real decodes when compared to the representational PD and PPVT models, as well as GLM models. DL models achieved approximately a 20% reduction in NMSE of decoded positions over tuning models and linear models (all $p < 10^{-3}$, Wilcoxon rank-sum test). Hence, DL models well encode kinematics in a manner that is more similar to the real neural data than linear encoders.

More specifically, we found that GLMh encoder achieved significantly better decoding performance than PD, PPVT, and GLM encoders, suggesting that incorporating kinematic history is important for reproducing decoding. This result is consistent with our observations that MLPh and RNN models outperformed the MLP, as shown in Fig. 7. As our prior results demonstrated that models incorporating kinematic history better reproduce dynamical aspects of motor cortical activity, this result is consistent with the observation that dynamical aspects of motor cortical activity are important for kinematic decoding [35], [36], [56]. Finally, while RNN models achieved better training loss and better reproduced PSTHs and population motifs, we did not find that the RNN model was always better than the MLPh model when decoding testing data.

D. Kinematics Decoded From DL and GLMh Models Better Match Closed-Loop Decoded Kinematics

As discussed earlier, BMIs are closed-loop systems where the user interacts with the decoded output, updating his or her motor

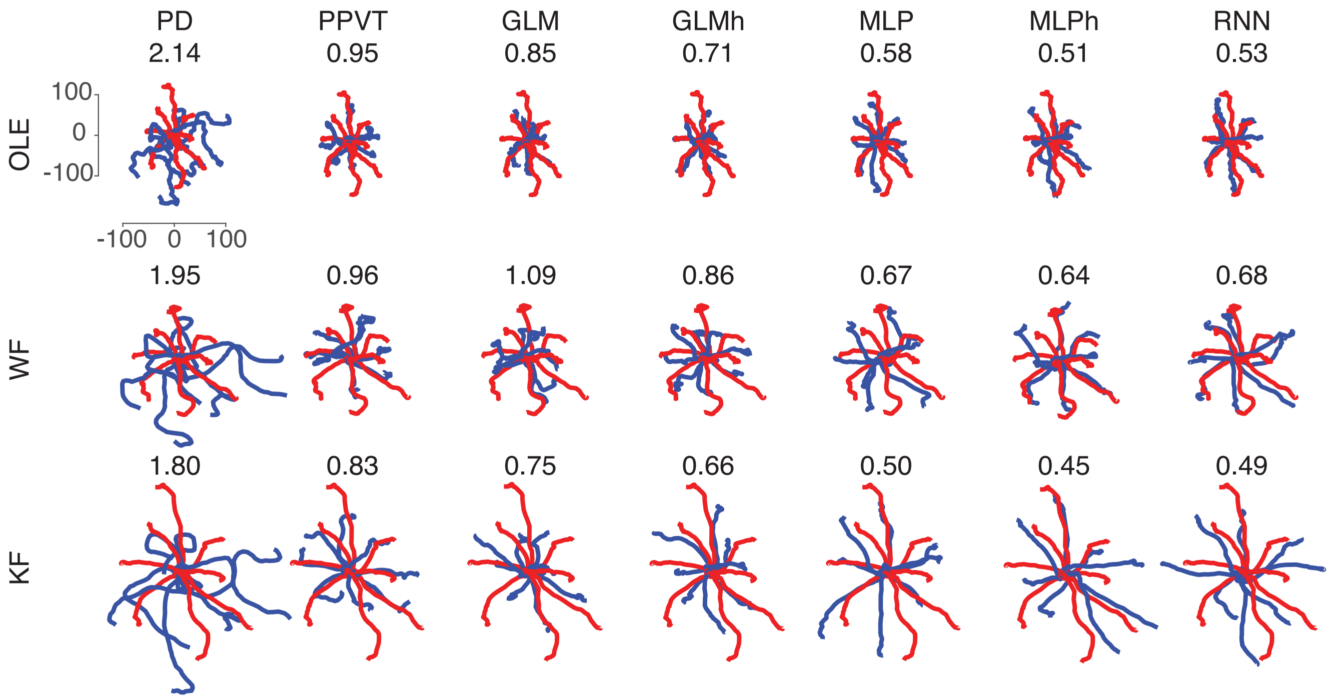

Fig. 7. Decoded movements from open-loop recorded and synthetic neural activity with three standard decoders (OLE, WF, and KF). For each direction, one randomly chosen trial is shown. In each panel, red lines represent decoded movements from real neural activity, and blue lines represent decoded movements from synthetic neural activity. NMSE of decoded positions are shown in each panel. Decoded positions from DL models have relatively better NMSE compared to tuning and linear models.

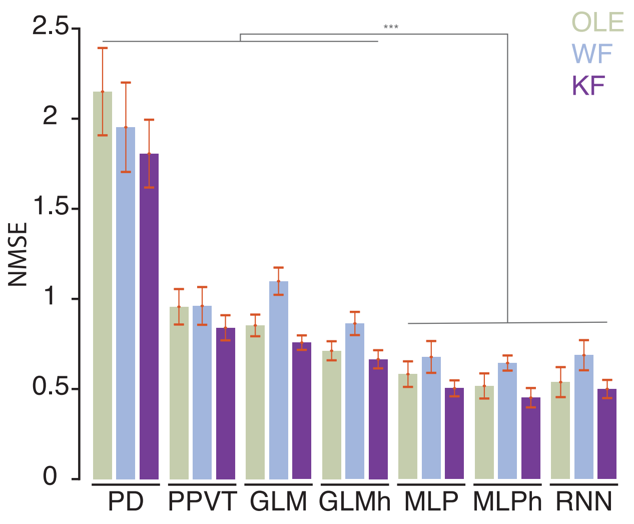

Fig. 8. NMSE of positions decoded from recorded neural activity and those decoded from synthetic neural activity in the open-loop testing dataset. The red vertical line on each bar is the standard deviation of NMSE of repeated experiments. DL models achieve lower value than tuning models across three decoders. *** denotes $p < 0.001$, Wilcoxon rank-sum test.

commands in response to visual feedback of the decoded output. Hence, the statistics of closed-loop neural population activity may differ from those during naturalistic reaching (e.g., [14], [24], [50], [51]). We therefore evaluated how well DL models generalized to reproduce closed-loop data, where monkeys update their motor commands in response to imperfect decoding. If DL models more faithfully reproduce prior decoded closed-loop kinematics, this is an additional evidence that they are more appropriate for closed-loop BMI use.

We trained encoding models on an open-loop center-out-and-back task performed with the native hand. However, we subsequently tested how well these encoding models could generate neural activity during BMI control from a previously collected closed-loop BMI dataset. We used the kinematics of the monkey's arm to generate synthetic neural activity, and subsequently decoded this activity post-hoc using the ReFIT-KF decoder. We then compared the post-hoc decoded movements to the empirical movements during closed-loop ReFIT-KF BMI control. These results are summarized in Fig. 9, demonstrating that even for closed-loop control, the DL encoding model synthetic activity could be better decoded to produce kinematics than tuning models. The DL model achieved a significant reduction ($>25\%$) in NMSE of decoded positions over PD and PPVT models (all $p < 10^{-3}$, Wilcoxon rank-sum test). Interestingly, we found that the GLMh model performed comparably to DL models, suggesting that this model may be appropriate for closed-loop BMI simulation. Together, these results indicate that DL models reproduce neural population activity better than tuning models when used in closed-loop settings, even though they were only trained in open-loop settings.

IV. Discussion

Prior work has demonstrated that motor cortical activity is heterogeneous, dynamic, and more complex than previously

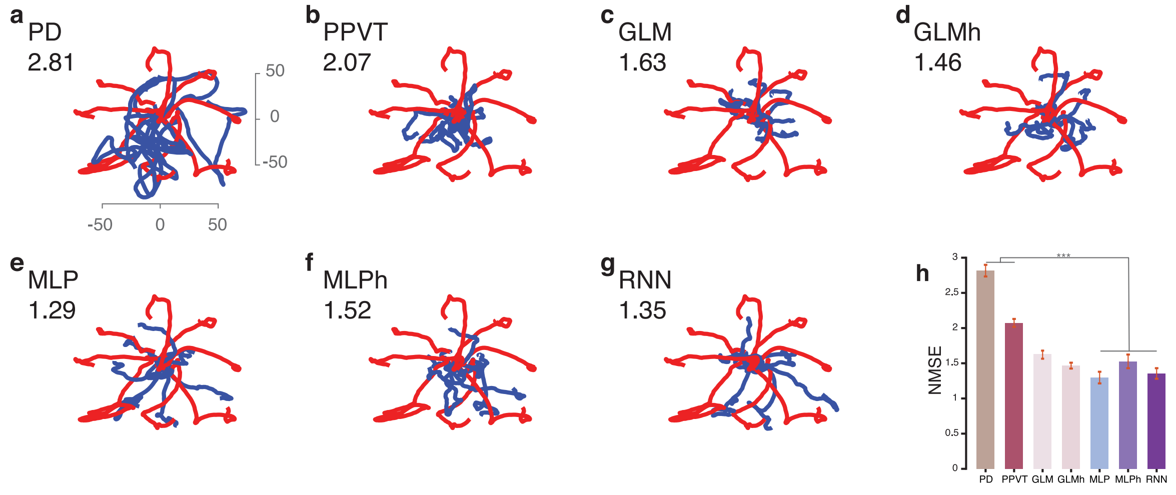

Fig. 9. Decoded movements from closed-loop recorded and synthetic neural activity with ReFIT-KF as decoder. In each panel, red lines are movements decoded from real neural data. Blue lines in each panel are movements decoded from synthetic neural activity of (a) PD, (b) PPVT, (c) GLM, (d) GLMh, (e) MLP, (f) MLPh, and (g) RNN, respectively. The average NMSE of each encoder is shown in each panel. (e) and (g) are decoded movements from MLP and RNN models, reflecting key motifs in original decodes and achieving the lowest NMSEs. (h) Normalized mean squared error (NMSE) between positions decoded from recorded neural activity and those decoded from synthetic neural activity in closed-loop dataset. *** denotes $p < 0.001$, Wilcoxon rank-sum test.

TABLE I
Summary of Monkey J's Results

		PD-v	PPVT-v	GLM-v	GLMh-v	GLM-pv	GLMh-pv	MLP-v	MLP-pv	MLPh-pv	RNN-pv
PSTH	Pearson's r	0.23	0.22	0.23	0.30	0.27	0.31	0.56	0.80	0.83	0.86
	MSE(spikes^2/s)	9.78	9.50	8.94	8.70	8.74	8.57	6.43	2.20	1.46	0.63
PCA	Pearson's r	0.29	0.28	0.34	0.45	0.41	0.47	0.67	0.87	0.90	0.94
	MSE	4.77	4.63	4.68	4.55	4.56	4.47	3.11	1.06	0.69	0.22
jPCA	R^2_{skew}	<0.01	<0.01	<0.01	0.27	0.05	0.28	0.08	0.22	0.25	0.32
	Skew Ratio	0.08	0.01	0.01	0.31	0.06	0.33	0.23	0.60	0.64	0.77
Decodes$_{OL}$	NMSE(OLE)	2.14	0.95	1.13	0.97	0.85	0.71	1.05	0.58	0.51	0.53
	NMSE(WF)	1.95	0.96	1.11	0.93	1.09	0.86	0.95	0.67	0.64	0.68
	NMSE(KF)	1.80	0.83	1.07	0.90	0.75	0.66	0.94	0.50	0.45	0.49
Decodes$_{CL}$	NMSE(ReFIT)	2.81	2.07	2.40	2.18	1.63	1.46	2.25	1.29	1.52	1.35

thought [55], [57], [87], [91]. Neural encoders that do not reproduce neural population statistics will limit the performance of BMI simulators. To overcome this limit, we employed high capacity, nonlinear neural network models to better capture complexities in the empirical data. Our results demonstrate that these models are better than traditional models in (1) reproducing heterogeneity in single electrode PSTHs, (2) reproducing nonlinear structure and dynamic features in the neural population, (3) matching decoded kinematics in open-loop experiments, and (4) matching decoded kinematics in closed-loop experiments. We summarize monkey J's results in Table I. Standard deviations are reported in Supplementary Table 1. Monkey L's results are summarized in Supplementary Tables 2 and 3. Note that we have not done an exhaustive search of hyperparameters, and it is likely that doing so could further improve our results. Future work can further optimize these hyperparameters.

We found that tuning and linear models significantly underestimated the variability of motor cortical neural activity. The PD model generated relatively static firing rates because it only represents the reach angle. The PPVT and GLM models incorporated kinematics, causing the PSTHs to have kinematic-resembling activity. In contrast, we found that DL models incorporating nonlinearity more closely reproduced empirical PSTHs. In addition to this, they also reproduced neural population motifs, resembling neural trajectories in PCA and jPCA projections.

When comparing to previously used tuning models, we found that decoding DL generated neural activity more closely resembled kinematics decoded from recorded activity, both in open-loop and closed-loop experiments. While DL models were superior to GLM and GLMh models in open-loop decoding, we observed that GLMh achieved similar NMSE to DL models in closed-loop decoding, as shown in Fig. 9. This is interesting, in light of the fact that GLMh was significantly worse than DL models in reproducing single electrode, neural population, and open-loop decodes. These results suggest that for closed-loop decoding, DL models and GLMh captures important kinematic variability in the neural activity. Further, this variability can be captured without accurately reproducing other neural features.

We found that incorporating kinematic history in neural encoders could improve neural encoding. A history over the inputs allows the output to have temporal structure, in that the same input at time t can produce a different output at time t based on the history of inputs at time $t-1$, $t-2$, and so forth. In this manner, the generated neural activity can have dynamics, which is an important and useful property of motor cortical activity [39], [87], [92]. However, we did find that DL models were best trained with an intermediate amount of history, or through an explicit dynamical model as in the RNN. Specifically, an MLPh model trained with 4 bins of input history performed better than when trained with 16 or 32 bins of input history. This suggests that although the MLPh can incorporate history over the inputs, it is difficult to model longer sequences for this application. This may be due to the fact that each input causes a commensurate increase in the number of parameters, making training more difficult. Hence, while MLPh can help model temporal structure in the output, it is limited in its ability to do so. The RNN, by implementing a dynamical system, can learn longer temporal dependencies. We emphasize though, that in open-loop data, while RNN models better reproduced PSTHs, PCA trajectories, and jPCA dynamics, we did not find that RNN is better than the MLP-based models when generalizing to closed-loop data (closed-loop NMSE not significantly different).

The input to DL neural encoders were 2D position and velocity. However, motor cortex also encodes additional kinematic [11] and kinetic variables including acceleration [93], [94], time [85], distance [95]–[99], joint angles [100], and forces [101], [102]. These additional kinematic parameters may improve neural encoders. To show that different inputs can impact performance, we trained MLP, GLM, and GLMh models with only a velocity input, and compared it to the same encoders trained with position and velocity inputs. We found that across these architectures, encoders using position and velocity (pv) generally matched or outperformed encoders using only velocity (v) as shown in Table I. We also tested DL encoders with acceleration as an additional input, but did not find that these encoders significantly increased performance. We note that, without position, MLP-v did not perform well. This answer is intuitive when considering that MLP-v does not incorporate history or dynamics, and therefore the same velocities will produce the same synthetic neural activity. When performing straight-line reaches, similar velocities will be encountered during the acceleration and deceleration of the reach. This causes synthetic neural activity in MLP-v to appear somewhat symmetric around the peak velocity, resulting in a neural encoder that does not capture key features of the neural population activity. Future work may improve encoding models by measuring and incorporating additional kinematic and kinetic variables as inputs. An important limitation of this simulation approach is that if a novel decoder algorithm incorporates newly discovered mechanisms or aspects of neural variance not captured by the inputs, then the simulator may not adequately evaluate this decoder's performance. This limits the scope of testable decoding algorithms to ones that operate on neural variability adequately captured by the available inputs.

The reported GLM models are linear Gaussian. We also tested a Poisson-GLM and Binomial-GLM, and found these encoders to perform significantly worse than DL models (data not shown). The GLM assumes the form of the error distribution between predicted and observed spike rates; however, these assumptions may not be valid. Further, GLMs with a nonlinear link function have nonlinearity at the output, but prior to the output, the model is linear. These nonlinear link functions have less capacity to model nonlinear relationships [5], [83] than neural networks. Therefore, DL models significantly improve upon linear and tuning methods in reconstructing single electrode and neural population activity, as well as decoding (except when comparing to GLMh, described above), showing that nonlinearity is an important factor in improving encoding models.

To demonstrate that DL models generalize for BMI simulators, we also assessed how well these models generated neural activity during closed-loop BMI experiments. Closed-loop BMI data has different statistics than offline testing data. However, if an encoding model generally captures how kinematics can be predictive of neural population activity, we would expect this encoding model to more accurately reproduce closed-loop BMI decoding performance. In closed-loop BMI decoding, the user's kinematics are drastically different; where as offline data comprises primarily ballistic reaches, closed-loop BMI decoding incorporates corrective movements, which are typically smaller in extent and fairly precise. We found that, when generalizing to closed-loop BMI kinematics, DL models and GLMh significantly outperformed tuning models, indicating that they may generalize better for closed-loop experiments. Future work should assess how these DL models perform in real-time closed-loop experiments. These experiments would utilize the neural encoder to generate synthetic neural activity and decode this activity in real-time.

There is a growing literature using deep learning in neuroscience. This body of work includes using variational autoencoders to denoise intracortical spike trains [56], and model early visual representations [65], sensory cortex [7], decision-making tasks [103]–[105], and motor tasks [9], [86], [87]. Prior studies training RNNs to perform motor tasks have used artificial units that resemble neurophysiological activity and dynamics [9], [86], [87]. However, these studies have important differences to this work. The goal of those studies is to train RNNs that reproduce animal behavior, so that the artificial RNN can be further analyzed for neuroscientific insight [104]. Along these lines, the inputs to these RNNs are task inputs and the outputs are behavior (such as electromyograms or kinematics). Further, in these RNNs, it is important for the artificial neurons to resemble the activity of empirically recorded neurons. In contrast, our study aimed to generate neural activity as a function of kinematics. As such, the inputs to our network are kinematics (as opposed to task inputs), and the outputs are neural activity (as opposed to EMG or kinematics). Finally, our work is not concerned about the activity of the hidden artificial units, as our goal is to reproduce neural activity at the output.

This work helps to improve BMI simulator tools for modeling closed-loop aspects of BMI decoding. Ultimately, the goal of this work is to accelerate the translation and design of BMI algorithms. Our work simulates monkey neural data, and not human neural data. We note that our approach could be applied to experiments with paralyzed participants: kinematics would correspond to imagined or intended kinematics, which match the movements of a cursor on a computer screen or a robotic arm (e.g., [19]–[21]) with simultaneous neural recordings. It is worth noting that results from monkey experiments have been translated to pilot clinical trials (e.g., [66]), showing the translational feasibility of performing optimization with monkey experiments.

V. Conclusion

Deep learning neural encoders are able to reproduce heterogeneous neural activity better than existing tuning models and GLM models. We showed that deep learning neural encoders are significantly better than prior neural encoders in reproducing PSTHs, neural population motifs, and matching decoded kinematics in open-loop and closed-loop data. Our results indicate that these deep learning neural encoders may significantly improve the fidelity of BMI simulators. Improving these simulators may facilitate developing, analyzing, and optimizing neural decoders to accelerate BMI clinical translation.

Acknowledgment

The authors would like to thank K. Shenoy for providing intracortical recordings analyzed in this paper; Stephen Ryu for surgical expertise; M. Risch, M. Wechsler, L. Yates, R. Steinbach, and S. Smith for surgical assistance and veterinary care; T. Senate, E. Castaneda, and B. Davis for administrative support. A GPU used for this research was donated by the NVIDIA Corporation.

References

[1] T. Gollisch and M. Meister, "Eye smarter than scientists believed: Neural computations in circuits of the retina," *Neuron*, vol. 65, pp. 150–164, Jan. 2010.

[2] A. D. Huberman *et al.*, "Architecture and activity-mediated refinement of axonal projections from a mosaic of genetically identified retinal ganglion cells," *Neuron*, vol. 59, pp. 425–438, Aug. 2008.

[3] J. W. Pillow *et al.*, "Spatio-temporal correlations and visual signalling in a complete neuronal population," *Nature*, vol. 454, pp. 995–999, Aug. 2008.

[4] N. C. Rust *et al.*, "Spatiotemporal elements of macaque v1 receptive fields," *Neuron*, vol. 46, pp. 945–956, Jun. 2005.

[5] L. T. McIntosh *et al.*, "Deep learning models of the retinal response to natural scenes," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, vol. 29, pp. 1369–1377.

[6] H. Wen *et al.*, "Neural encoding and decoding with deep learning for dynamic natural vision," *Cerebral Cortex*, vol. 28, pp. 4136–4160, Dec. 2018.

[7] D. L. K. Yamins and J. J. DiCarlo, "Using goal-driven deep learning models to understand sensory cortex," *Nature Neurosci.*, vol. 19, pp. 356–365, Mar. 2016.

[8] M. M. Churchland *et al.*, "Neural population dynamics during reaching," *Nature*, vol. 487, pp. 51–56, Jul. 2012.

[9] J. A. Michaels, B. Dann, and H. Scherberger, "Neural population dynamics during reaching are better explained by a dynamical system than representational tuning," *PLoS Comput. Biol.*, vol. 12, no. 11, 2016, Art. no. e1005175.

[10] R. E. Hampson *et al.*, "Developing a hippocampal neural prosthetic to facilitate human memory encoding and recall," *J. Neural Eng.*, vol. 15, Jun. 2018, Art. no. 036014.

[11] S. J. Bensmaia and L. E. Miller, "Restoring sensorimotor function through intracortical interfaces: Progress and looming challenges," *Nature Rev. Neurosci.*, vol. 15, pp. 313–325, May 2014.

[12] V. Gilja *et al.*, "Challenges and opportunities for next-generation intracortically based neural prostheses," *IEEE Trans. Biomed. Eng.*, vol. 58, no. 7, pp. 1891–1899, Jul. 2011.

[13] J. E. O'Doherty *et al.*, "Active tactile exploration using a brain-machine-brain interface," *Nature*, vol. 479, pp. 228–231, Oct. 2011.

[14] J. P. Cunningham *et al.*, "A closed-loop human simulator for investigating the role of feedback control in brain-machine interfaces," *J. Neurophysiol.*, vol. 105, pp. 1932–1949, Apr. 2011.

[15] K. D. Anderson, "Targeting recovery: Priorities of the spinal cord-injured population," *J. Neurotrauma*, vol. 21, pp. 1371–1383, Oct. 2004.

[16] B. S. Armour *et al.*, "Prevalence and causes of Paralysis-United states, 2013," *Amer. J. Public Health*, vol. 106, pp. 1855–1857, Oct. 2016.

[17] C. & Dana Reeve Foundation, *One Degree of Separation: Paralysis and Spinal Cord Injury in the United States*. Short Hills, NJ, USA: Christopher & Dana Reeve Foundation, 2009.

[18] A. B. Ajiboye *et al.*, "Restoration of reaching and grasping movements through brain-controlled muscle stimulation in a person with tetraplegia: A proof-of-concept demonstration," *Lancet*, vol. 389, pp. 1821–1830, May 2017.

[19] J. L. Collinger *et al.*, "High-performance neuroprosthetic control by an individual with tetraplegia," *Lancet*, vol. 381, pp. 557–564, Feb. 2013.

[20] L. R. Hochberg *et al.*, "Neuronal ensemble control of prosthetic devices by a human with tetraplegia," *Nature*, vol. 442, pp. 164–171, Jul. 2006.

[21] L. R. Hochberg *et al.*, "Reach and grasp by people with tetraplegia using a neurally controlled robotic arm," *Nature*, vol. 485, pp. 372–375, May 2012.

[22] C. Pandarinath *et al.*, "High performance communication by people with paralysis using an intracortical brain-computer interface," *eLife*, vol. 6, Feb. 2017, Art. no. e18554, doi: 10.7554/eLife.18554.

[23] J. Dethier *et al.*, "Design and validation of a real-time spiking-neural-network decoder for brain-machine interfaces," *J. Neural Eng.*, vol. 10, Jun. 2013, Art. no. 036008.

[24] J. C. Kao *et al.*, "Information systems opportunities in brain machine interface decoders," *Proc. IEEE*, vol. 102, no. 5, pp. 666–682, May 2014.

[25] P. R. Kennedy *et al.*, "Direct control of a computer from the human central nervous system," *IEEE Trans. Rehabil. Eng.*, vol. 8, no. 2, pp. 198–202, Jun. 2000.

[26] D. M. Taylor, S. I. H. Tillery, and A. B. Schwartz, "Direct cortical control of 3D neuroprosthetic devices," *Science*, vol. 296, pp. 1829–1832, Jun. 2002.

[27] J. M. Carmena *et al.*, "Learning to control a brain-machine interface for reaching and grasping by primates," *PLoS Biol.*, vol. 1, Nov. 2003, Art. no. E42.

[28] P. Nuyujukian *et al.*, "A nonhuman primate brain–computer typing interface," *Proc. IEEE*, vol. 105, no. 1, pp. 66–72, Jan. 2017.

[29] M. Velliste *et al.*, "Cortical control of a prosthetic arm for self-feeding," *Nature*, vol. 453, pp. 1098–1101, Jun. 2008.

[30] C. E. Bouton *et al.*, "Restoring cortical control of functional movement in a human with quadriplegia," *Nature*, vol. 533, pp. 247–250, May 2016.

[31] P. Nuyujukian *et al.*, "Cortical control of a tablet computer by people with paralysis," *PLoS One*, vol. 13, Nov. 2018, Art. no. e0204566.

[32] S. I. Ryu and K. V. Shenoy, "Human cortical prostheses: Lost in translation?," *Neurosurgical Focus*, vol. 27, 2009, Art. no. E5.

[33] V. Gilja *et al.*, "A high-performance neural prosthesis enabled by control algorithm design," *Nature Neurosci.*, vol. 15, pp. 1752–1757, Dec. 2012.

[34] M. M. Shanechi *et al.*, "Rapid control and feedback rates enhance neuroprosthetic control," *Nature Commun.*, vol. 8, 2017, Art. no. 13825.

[35] M. Aghagolzadeh and W. Truccolo, "Inference and decoding of motor cortex low-dimensional dynamics via latent state-space models," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 24, no. 2, pp. 272–282, Feb. 2016.

[36] J. C. Kao *et al.*, "Single-trial dynamics of motor cortex and their applications to brain-machine interfaces," *Nature Commun.*, vol. 6, Jul. 2015, Art. no. 7759.

[37] J. C. Kao, S. I. Ryu, and K. V. Shenoy, "Leveraging neural dynamics to extend functional lifetime of brain-machine interfaces," *Sci. Rep.*, vol. 7, Aug. 2017, Art. no. 7395.

[38] E. L. Dyer *et al.*, "A cryptography-based approach for movement decoding," *Nature Biomed. Eng.*, vol. 1, pp. 967–976, Dec. 2017.

[39] C. Pandarinath *et al.*, "Latent factors and dynamics in motor cortex and their application to Brain–Machine interfaces," *J. Neurosci.*, vol. 38, pp. 9390–9401, Oct. 2018.

[40] V. Lawhern *et al.*, "Population decoding of motor cortical activity using a generalized linear model with hidden states," *J. Neurosci. Methods*, vol. 189, pp. 267–280, Jun. 2010.

[41] M. Aghagolzadeh and W. Truccolo, "Latent state-space models for neural decoding," in *Proc. IEEE Conf. Eng. Med. Biol. Soc.*, 2014, vol. 2014, pp. 3033–3036.

[42] H. Abbaspourazad *et al.*, "Identifying multiscale hidden states to decode behavior," in *Proc. IEEE Conf. Eng. Med. Biol. Soc.*, Jul. 2018, vol. 2018, pp. 3778–3781.

[43] J. H. Macke *et al.*, "Empirical models of spiking in neural populations," in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, Eds., Red Hook, NY, USA: Curran Associates, Inc., 2011, pp. 1350–1358.

[44] J. E. Kulkarni and L. Paninski, "Common-input models for multiple neural spike-train data," *Network*, vol. 18, pp. 375–407, Dec. 2007.

[45] D. Sussillo *et al.*, "Making brain–machine interfaces robust to future neural variability," *Nature Commun.*, vol. 7, Dec. 2016, Art. no. 13749.

[46] J. M. Fan *et al.*, "Intention estimation in brain-machine interfaces," *J. Neural Eng.*, vol. 11, Feb. 2014, Art. no. 016004.

[47] K. Ganguly and J. M. Carmena, "Emergence of a stable cortical map for neuroprosthetic control," *PLoS Biol.*, vol. 7, Jul. 2009, Art. no. e1000153.

[48] K. Ganguly *et al.*, "Reversible large-scale modification of cortical networks during neuroprosthetic control," *Nature Neurosci.*, vol. 14, pp. 662–667, May 2011.

[49] K. V. Shenoy and J. M. Carmena, "Combining decoder design and neural adaptation in brain-machine interfaces," *Neuron*, vol. 84, no. 4, pp. 665–680, 2014.

[50] S. M. Chase, A. B. Schwartz, and R. E. Kass, "Bias, optimal linear estimation, and the differences between open-loop simulation and closed-loop performance of spiking-based brain-computer interface algorithms," *Neural Netw.*, vol. 22, no. 9, pp. 1203–1213, 2009.

[51] S. Koyama *et al.*, "Comparison of brain-computer interface decoding algorithms in open-loop and closed-loop control," *J. Comput. Neurosci.*, vol. 29, pp. 73–87, Aug. 2010.

[52] F. R. Willett *et al.*, "Feedback control policies employed by people using intracortical brain-computer interfaces," *J. Neural Eng.*, vol. 14, Feb. 2017, Art. no. 016001.

[53] F. R. Willett *et al.*, "Principled BCI decoder design and parameter selection using a feedback control model," *Sci. Rep.*, vol. 9, Jun. 2019, Art. no. 8881.

[54] A. P. Georgopoulos *et al.*, "On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex," *J. Neurosci.*, vol. 2, pp. 1527–1537, Nov. 1982.

[55] M. M. Churchland and K. V. Shenoy, "Temporal complexity and heterogeneity of single-neuron activity in premotor and motor cortex," *J. Neurophysiol.*, vol. 97, pp. 4235–4257, 2007.

[56] C. Pandarinath *et al.*, "Inferring single-trial neural population dynamics using sequential auto-encoders," *Nature Methods*, vol. 15, pp. 805–815, Oct. 2018.

[57] M. M. Churchland *et al.*, "Neural population dynamics during reaching," *Nature*, vol. 487, pp. 51–56, Jul. 2012.

[58] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., Red Hook, NY, USA: Curran Associates, Inc., 2012, pp. 1097–1105.

[59] C. Szeged *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2015, pp. 1–9.

[60] K. He *et al.*, "Identity mappings in deep residual networks," in *Computer Vision* (Lecture Notes in Computer Science). Berlin, Germany: Springer, Oct. 2016, pp. 630–645.

[61] G. Cheng, P. Zhou, and J. Han, "Learning rotation-Invariant convolutional neural networks for object detection in VHR optical remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 12, pp. 7405–7415, Dec. 2016.

[62] J. Han *et al.*, "Object detection in optical remote sensing images based on weakly supervised learning and high-level feature learning," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 6, pp. 3325–3337, Jun. 2015.

[63] D. Zhang *et al.*, "Detection of co-salient objects by looking deep and wide," *Int. J. Comput. Vis.*, vol. 120, pp. 215–232, Nov. 2016.

[64] J. Han *et al.*, "Background prior-based salient object detection via deep reconstruction residual," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 8, pp. 1309–1321, Aug. 2015.

[65] J. Lindsey *et al.*, "The effects of neural resource constraints on early visual representations," in *Proc. Int. Conf. Learn. Representations*, 2019.

[66] V. Gilja *et al.*, "Clinical translation of a high-performance neural prosthesis," *Nature Med.*, vol. 21, no. 10, pp. 1142–1145, 2015.

[67] C. A. Chestek *et al.*, "Long-term stability of neural prosthetic control signals from silicon cortical arrays in rhesus macaque motor cortex," *J. Neural Eng.*, vol. 8, Aug. 2011, Art. no. 045005.

[68] P. Nuyujukian *et al.*, "Monkey models for brain-machine interfaces: The need for maintaining diversity," *Conf. IEEE Eng. Med. Biol. Soc.*, vol. 2011, pp. 1301–1305, Jan. 2011.

[69] B. M. Yu *et al.*, "Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity," *J. Neurophysiol.*, vol. 102, pp. 614–635, Jul. 2009.

[70] K. V. Shenoy, M. Sahani, and M. M. Churchland, "Cortical control of arm movements: A dynamical systems perspective," *Annu. Rev. Neurosci.*, vol. 36, pp. 337–359, Jul. 2013.

[71] J. P. Cunningham and B. M. Yu, "Dimensionality reduction for large-scale neural recordings," *Nature Neurosci.*, vol. 17, pp. 1500–1509, Aug. 2014.

[72] C. Pandarinath *et al.*, "Latent factors and dynamics in motor cortex and their application to Brain-Machine interfaces," *J. Neurosci.*, vol. 38, pp. 9390–9401, Oct. 2018.

[73] E. Salinas and L. F. Abbott, "Vector reconstruction from firing rates," *J. Comput. Neurosci.*, vol. 1, pp. 89–107, Jun. 1994.

[74] J. M. Carmena *et al.*, "Learning to control a Brain–Machine interface for reaching and grasping by primates," *PLoS Biol.*, vol. 1, Oct. 2003, Art. no. e42.

[75] S.-P. Kim *et al.*, "Neural control of computer cursor velocity by decoding motor cortical spiking activity in humans with tetraplegia," *J. Neural Eng.*, vol. 5, pp. 455–476, Dec. 2008.

[76] W. Wu *et al.*, "Neural decoding of cursor motion using a Kalman filter," in *Advances in Neural Information Processing Systems 15*, S. Becker, S. Thrun, and K. Obermayer, Eds., Cambridge, MA, USA: MIT Press, 2003, pp. 133–140.

[77] E. Stark and M. Abeles, "Predicting movement from multiunit activity," *J. Neurosci.*, vol. 27, pp. 8387–8394, Aug. 2007.

[78] G. W. Fraser *et al.*, "Control of a brain-computer interface without spike sorting," *J. Neural Eng.*, vol. 6, Oct. 2009, Art. no. 055004.

[79] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2014.

[80] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, Mar. 2010, pp. 249–256.

[81] S. Gerwinn *et al.*, "Bayesian inference for generalized linear models for spiking neurons," *Frontiers Comput. Neurosci.*, vol. 4, p. 12, May 2010, doi: 10.3389/fncom.2010.00012.

[82] W. Truccolo., "A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects," *J. Neurophysiol.*, vol. 93, pp. 1074–1089, Feb. 2005.

[83] A. S. Benjamin *et al.*, "Modern machine learning as a benchmark for fitting neural responses," *Front. Comput. Neurosci.*, vol. 12, p. 56, Jul. 2018, doi: 10.3389/fncom.2018.00056.

[84] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, Nov. 2016.

[85] M. T. Kaufman *et al.*, "The largest response component in motor cortex reflects movement timing but not movement type," *eNeuro*, vol. 3, no. 4, Aug. 2016, doi: 10.1523/ENEURO.0085-16.

[86] G. Hennequin, T. P. Vogels, and W. Gerstner, "Optimal control of transient dynamics in balanced networks supports generation of complex movements," *Neuron*, vol. 82, pp. 1394–1406, Jun. 2014.

[87] D. Sussillo *et al.*, "A neural network that finds a naturalistic solution for the production of muscle activity," *Nature Neurosci.*, vol. 18, no. 7, pp. 1025–1033, 2015.

[88] C. Pandarinath *et al.*, "Neural population dynamics in human motor cortex during movements in people with ALS," *Elife*, vol. 4, Jun. 2015, Art. no. e07436.

[89] G. F. Elsayed and J. P. Cunningham, "Structure in neural population recordings: An expected byproduct of simpler phenomena?," *Nature Neurosci.*, vol. 20, pp. 1310–1318, Sep. 2017.

[90] N. Even-Chen *et al.*, "Augmenting intracortical brain-machine interface with neurally driven error detectors," *J. Neural Eng.*, vol. 14, Dec. 2017, Art. no. 066007.

[91] M. M. Churchland *et al.*, "Cortical preparatory activity: Representation of movement or first cog in a dynamical machine?," *Neuron*, vol. 68, pp. 387–400, Nov. 2010.

[92] A. A. Russo *et al.*, "Motor cortex embeds muscle-like commands in an untangled population response," *Neuron*, vol. 97, pp. 953–966, Feb. 2018.

[93] W. Wu *et al.*, "Neural decoding of cursor motion using a kalman filter," in *Advances in Neural Information Processing Systems 15*, S. Becker, S. Thrun, and K. Obermayer, Eds., Cambridge, MA, USA: MIT Press, 2003, pp. 133–140.

[94] L. Paninski *et al.*, “Spatiotemporal tuning of motor cortical neurons for hand position and velocity,” *J. Neurophysiol.*, vol. 91, pp. 515–532, Jan. 2004.

[95] M. M. Churchland, G. Santhanam, and K. V. Shenoy, “Preparatory activity in premotor and motor cortex reflects the speed of the upcoming reach,” *J. Neurophysiol.*, vol. 96, pp. 3130–3146, Dec. 2006.

[96] Q. G. Fu, J. I. Suarez, and T. J. Ebner, “Neuronal specification of direction and distance during reaching movements in the superior precentral premotor area and primary motor cortex of monkeys,” *J. Neurophysiol.*, vol. 70, pp. 2097–2116, Nov. 1993.

[97] J. Messier and J. F. Kalaska, “Covariation of primate dorsal premotor cell activity with direction and amplitude during a memorized-delay reaching task,” *J. Neurophysiol.*, vol. 84, pp. 152–165, Jul. 2000.

[98] D. W. Moran and A. B. Schwartz, “Motor cortical representation of speed and direction during reaching,” *J. Neurophysiol.*, vol. 82, pp. 2676–2692, Nov. 1999.

[99] A. Riehle and J. Requin, “Monkey primary motor and premotor cortex: Single-cell activity related to prior information about direction and extent of an intended movement,” *J. Neurophysiol.*, vol. 61, pp. 534–549, Mar. 1989.

[100] J. A. Pruszynski *et al.*, “Primary motor cortex underlies multi-joint integration for fast feedback control,” *Nature*, vol. 478, pp. 387–390, Sep. 2011.

[101] D. R. Humphrey, E. M. Schmidt, and W. D. Thompson, “Predicting measures of motor performance from multiple cortical spike trains,” *Science*, vol. 170, pp. 758–762, Nov. 1970.

[102] M. Hepp-Reymond *et al.*, “Context-dependent force coding in motor and premotor cortical areas,” *Exp. Brain Res.*, vol. 128, pp. 123–133, Sep. 1999.

[103] V. Mante *et al.*, “Context-dependent computation by recurrent dynamics in prefrontal cortex,” *Nature*, vol. 503, pp. 78–84, Nov. 2013.

[104] H. F. Song, G. R. Yang, and X. J. Wang, “Training excitatory-inhibitory recurrent neural networks for cognitive tasks: A simple and flexible framework,” *PLoS Comput. Biol.*, vol. 12, no. 2, pp. 1–30, 2016.

[105] W. Chaisangmongkon *et al.*, “Computing by robust transience: How the fronto-parietal network performs sequential, category-based decisions,” *Neuron*, vol. 93, no. 6, pp. 1504–1517, 2017.