Improved Spike-based Brain-Machine Interface Using Bayesian Adaptive Kernel Smoother and Deep Learning

NUR AHMADI[1,2], (Graduate Student Member, IEEE), TIMOTHY G. CONSTANDINOU[1,2,3], (SENIOR MEMBER, IEEE), AND CHRISTOS-SAVVAS BOUGANIS[1], (Senior Member, IEEE)

[1]Department of Electrical and Electronic Engineering, Imperial College London, London, SW7 2AZ, UK
[2]Centre for Bio-Inspired Technology, Institute of Biomedical Engineering, Imperial College London, London, SW7 2AZ, UK
[3]Care Research and Technology Centre at the UK Dementia Research Institute (UKDRI)

Corresponding author: Nur Ahmadi (e-mail: n.ahmadi16@imperial.ac.uk).

This work was supported in part by the Engineering and Physical Sciences Research Council (EPSRC) Award (No. EP/M020975/1) and in part by the Indonesia Endowment Fund for Education (LPDP) Award (No. PRJ-123/LPDP/2016).

ABSTRACT Multiunit activity (MUA) has been proposed to mitigate the robustness issue faced by single-unit activity (SUA)-based brain-machine interfaces (BMIs). Most MUA-based BMIs still employ a binning method for extracting firing rates and linear decoder for decoding behavioural parameters. The limitations of binning and linear decoder lead to suboptimal performance of MUA-based BMIs. To address this issue, we propose Bayesian adaptive kernel smoother (BAKS) as the feature extraction method and long short-term memory (LSTM)-based deep learning as the decoding algorithm. We evaluated the proposed methods for reconstructing (offline) hand kinematics from intracortical neural data chronically recorded from the motor cortex of a monkey. Experimental results showed that BAKS coupled with LSTM outperformed other combinations of feature extraction method (binning or fixed kernel smoother) and decoding algorithm (Kalman filter or Wiener filter). Overall results demonstrate the effectiveness of BAKS and LSTM for improving the decoding performance of MUA-based BMIs.

KEYWORDS Brain-machine interface, Bayesian adaptive kernel smoother, deep learning, firing rate, multiunit activity.

I. INTRODUCTION

BRAIN-machine interfaces (BMIs) seek to restore lost motor function in severely paralysed patients by translating brain activity into control signals for guiding assistive devices. Numerous BMI studies utilising neuronal action potentials or spikes —also known as single-unit activity (SUA)— have shown compelling results in animals [1]–[4] and humans [5]–[8]. Nevertheless, spike recordings are not chronically stable, and the number of observable units progressively decline over time [9]–[11]. Several factors thought to cause this instability are glial scar formation (induced by neural tissue responses) encapsulating the electrodes, micromotion of the electrodes, insulation degradation, and mechanical breakage [12], [13]. The instability of SUA hinders clinical translation of SUA-based BMIs.

To overcome the above problem, multiunit activity (MUA) has been proposed as an alternative input signal to single-unit activity (SUA) [11], [14]–[16]. MUA refers to all spikes detected via a threshold-crossing technique without classifying (sorting) further into individual units. Thus, MUA represents the aggregate spikes from an ensemble of neurons in the vicinity of the recording electrode tip. Compared to SUA, MUA is more stable and requires simpler signal processing. Most MUA-based BMIs employ a binning method for extracting features (firing rates) and linear decoders for decoding movement parameters [17]–[19]. Binning estimates firing rates by counting the number of spikes within a predefined bin/window width. Despite being simple and fast, binning results in a coarse/noisy estimate of firing rates. As movements are usually smooth or continuous over time, a method that can yield a smooth estimate of firing rates could potentially improve decoding performance of MUA-based BMIs. Additionally, the use of linear decoders with their inherent assumptions leads to suboptimal decoding performance since neural signals often exhibit non-linear, non-stationary, and non-Gaussian characteristics [20]. Therefore, it is highly desirable to develop a decoding algorithm that is robust against the above neural signal characteristics. The rise of deep learning in recent years has presented the opportunity to potentially improve the decoding performance of MUA-based BMIs. So far, however, there have been very few studies utilising deep learning for MUA-based BMIs.

The present work aims to improve the decoding performance of MUA-based BMIs by simultaneously proposing (1) Bayesian adaptive kernel smoother (BAKS) originally reported in [21] as the firing rate estimation method and (2) long short-term memory (LSTM)-based deep learning as the decoding algorithm. We evaluate BAKS and LSTM for decoding (offline) hand kinematics from neural signals chronically recorded from the primary motor cortex (M1) area of a monkey while performing self-paced reaching tasks. We benchmark the proposed methods against two firing rate estimation methods (binning and fixed kernel smoother) and two decoding algorithms (Kalman filter and Wiener filter) as reported in previous studies (e.g. [16], [22]–[24]). Lastly, we compare the decoding performance between MUA and SUA using the proposed methods.

The remainder of this paper is organised as follows: Section II describes the methods, including experimental setup, signal processing, and decoding algorithm; Section III presents the empirical results, followed by analyses and discussion in Section IV; and finally, the conclusion is drawn in Section V.

II. METHODS

Fig. 1 illustrates the schematic overview of spike-based BMI system. Firing rate estimation was performed in an overlapping fashion to match the timescale of the kinematic data. The estimated firing rates were standardised (i.e. z-transformed) before being fed to the decoding algorithm. All the experiments and analyses were done in Python.

A. NEURAL RECORDINGS

Neural data were recorded from the primary motor cortex (M1) area of an adult male Rhesus macaque monkey (*Macaca mulatta*) by Sabes lab [25]. The recordings were made with a 96-channel Utah microelectrode array (platinum contact, 400 kΩ impedance, 400 μm interelectrode spacing, 1 mm electrode length) referenced to a silver wire placed under the dura (several cm away from the electrodes). The recordings were preamplified and filtered by using a 4th-order low-pass filter at 7.5 kHz and were then digitised with 16-bit resolution at 24.4 kHz sampling rate. These digitised recordings are referred to as raw neural signals. Details of the experimental setup are described elsewhere [26]. We used a total of 26 recording sessions spanning 7.3 months between the first (I20160627_01) and last (I20170131_02) sessions with a varying duration from 6 to 13.6 minutes (average of 8.88 $\pm$ 1.96 minutes). The first and last sessions correspond to 110 and 328 days after the electrode implantation date, respectively.

B. BEHAVIOURAL TASK

The monkey was trained to reach randomly drawn circular targets which were uniformly distributed around an 8×8 square grid. The target was acquired when the monkey reached the target using his fingertip and held it for 450 ms. Upon every target acquisition, a new random target was presented immediately without an inter-trial interval. The fingertip position of the reaching hand and the target position (in x–y Cartesian coordinates with mm unit) were both sampled at 250 Hz. The position data were then low pass filtered with a non-causal, 4th-order Butterworth filter at 10 Hz to reject sensor noise. Velocity and acceleration data were computed using the first and second derivative of the position data. A more detailed description of the behavioural task is given in [26].

C. SPIKE SIGNAL PROCESSING

1) Multiunit activity (MUA)

The raw neural signals were band-pass filtered with a causal, 4th-order Butterworth IIR filter between 500 to 5000 Hz. Spikes were then detected whenever the absolute value of the band-passed signals crossed a threshold value (typically set to between 3.5 and 4.0 times the standard deviation). Here, MUA refers to all the detected spikes aggregated per channel. Only MUA with spike rates exceeding 0.5 Hz were included. The number of MUA varied from 77 to 95 units (average of 87.08 $\pm$ 4.44) across 26 recording sessions.

2) Single-unit activity (SUA)

SUA was obtained by sorting (i.e. classifying) the detected spikes into distinct putative single units via principal component analysis and template matching. Each channel could contain more than one units. More detailed information on the spike detection and sorting processes can be found in [26]. We Only included SUA with spike rates above 0.5 Hz, which resulted in a varying number of SUA ranging from 91 to 157 units (average of 125.73 ± 13.95) across 26 recording sessions.

D. FIRING RATE ESTIMATION

1) Bayesian Adaptive Kernel Smoother (BAKS)

BAKS estimates firing rate, $\hat{\lambda}(t)$, by convolving a spike train $\rho(t)$ with a kernel function with adaptive bandwidth $K_{h(t)}(t)$,

$$\hat{\lambda}(t) = \int_{-\infty}^{\infty} K_{h(t)}(\tau)\rho(t-\tau)d\tau \tag{1}$$

where $h(t)$ denotes adaptive bandwidth parameter. The spike train is formulated as

$$\rho(t) = \sum_{i=1}^{n} \delta(t - t_i) \tag{2}$$

where t_i represents the spike time, n is the number of spikes, and $\delta(t)$ is Diract function. By substituting Eq. (2) into Eq. (1), we obtain

$$\hat{\lambda}(t) = \sum_{i=1}^{n} K_{h(t)}(t - t_i) \tag{3}$$

We proposed a Gaussian kernel function and Gamma prior distribution on the precision parameter ($\sigma(t)$), where $\sigma(t) = 1/h(t)^2$. Due to conjugate prior relationship, this

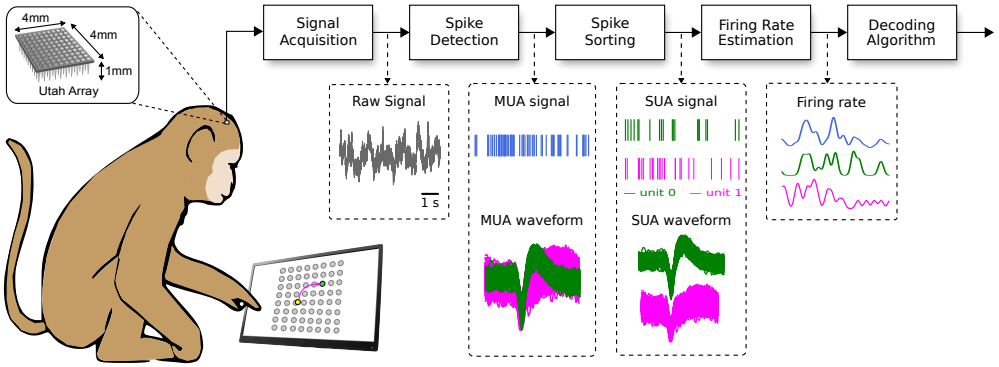

FIGURE 1. Schematic overview of spike-based BMI system

combination leads to an analytical expression of the posterior distribution. The Gamma prior distribution is given by

$$\pi(\sigma(t)) = \frac{\sigma(t)^{\alpha-1}}{\Gamma(\alpha)\beta^{\alpha}} \exp\left\{-\frac{\sigma(t)}{\beta}\right\}, \quad \sigma > 0 \tag{4}$$

where $\alpha > 0$ is the shape parameter, $\beta > 0$ is the scale parameter, and $\Gamma(\alpha)$ is Gamma function. By the change-of-variable formula and transformation technique, Eq. (4) can be expressed as:

$$\pi(h(t)) = \frac{2h(t)^{-2\alpha-1}}{\Gamma(\alpha)\beta^{\alpha}} \exp\left\{-\frac{1}{\beta h(t)^2}\right\} \tag{5}$$

Using Bayes' theorem, the posterior distribution of kernel bandwidth, $\pi(h(t)|\rho(t))$, can be computed by

$$\pi(h(t)|\rho(t)) = \frac{\hat{f}(\rho(t)|h(t))\pi(h(t))}{\int \hat{f}(\rho(t)|h(t))\pi(h(t))dh(t)} \tag{6}$$

where $\hat{f}(\rho(t)|h(t))$ represents an approximation of likelihood function based on Gaussian kernel, which is formulated as

$$\hat{f}(\rho(t)|h(t)) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\sqrt{2\pi}h(t)} \exp\left\{-\frac{(t-t_i)^2}{2h(t)^2}\right\} \tag{7}$$

By substituting Eq. (5) and Eq. (7) into Eq. (6), we obtain an analytical expression as follows

$$\pi(h(t)|\rho(t)) = \frac{\sum_{i=1}^{n} h(t)^{-2\alpha-2} \exp\left\{-\frac{1}{h(t)^2}\left[\frac{(t-t_i)^2}{2} + \frac{1}{\beta}\right]\right\}}{\frac{1}{2}\Gamma(\alpha+\frac{1}{2})\sum_{i=1}^{n}\left[\frac{(t-t_i)^2}{2} + \frac{1}{\beta}\right]^{(-\alpha-\frac{1}{2})}} \tag{8}$$

Under squared error loss function, the adaptive bandwidth estimate can be computed by

$$\hat{h}(t) = \frac{\Gamma(\alpha)\sum_{i=1}^{n}\left[\frac{(t-t_i)^2}{2} + \frac{1}{\beta}\right]^{-\alpha}}{\Gamma(\alpha+\frac{1}{2})\sum_{i=1}^{n}\left[\frac{(t-t_i)^2}{2} + \frac{1}{\beta}\right]^{-\alpha-\frac{1}{2}}} \tag{9}$$

The adaptive bandwidth in Eq. (9) is then used for firing rate estimation in Eq. (3).

Firing rate was estimated using 256 ms window width, parameter shape (α) = 4, and parameter scale (β) = $n^{4/5}$; n represents the number of spikes. The formula of BAKS and its parameter setting are described in [21].

2) Fixed kernel smoother (FKS)

FKS estimates firing rate by convolving a spike train with a 256 ms Gaussian kernel with fixed bandwidth parameter (110 ms). This bandwidth value was chosen because it resulted in better performance than 50 ms and 150 ms used in [22].

3) Binning

Binning estimates firing rate by computing the number of spikes within a 256 ms rectangular window. As in BAKS and FKS, the window width was set to 256 ms because, according to our previous study [27], this value yielded the highest decoding performance.

E. DECODING ALGORITHM

1) Long short-term memory (LSTM)

LSTM, proposed by Hochreiter and Schmidhuber in 1997 [28], is one of the most popular deep learning methods and has achieved state-of-the-art performance in various tasks, particularly those with time-series data [29]. It has successfully addressed the vanishing gradient problem commonly encountered in traditional recurrent neural networks (RNNs). LSTMs can effectively learn long-term temporal dependencies via a memory cell that maintains its state overtime and gating mechanism that controls the flow of information into and out of the memory cell. The states of LSTM components at timestep t are mathematically expressed as follows:

$$\begin{aligned}
\mathbf{f}_t &= \sigma(\mathbf{W}_f\mathbf{x}_t + \mathbf{U}_f\mathbf{h}_{t-1} + \mathbf{b}_f) \\
\mathbf{i}_t &= \sigma(\mathbf{W}_i\mathbf{x}_t + \mathbf{U}_i\mathbf{h}_{t-1} + \mathbf{b}_i) \\
\tilde{\mathbf{c}}_t &= \tanh(\mathbf{W}_c\mathbf{x}_t + \mathbf{U}_c\mathbf{h}_{t-1} + \mathbf{b}_c) \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o\mathbf{x}_t + \mathbf{U}_o\mathbf{h}_{t-1} + \mathbf{b}_o) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t)
\end{aligned} \tag{10}$$

TABLE 1. Hyperparameter configuration of SUA- and MUA-driven LSTM decoders across firing rate estimation methods

Hyperparameter	Value range	SUA			MUA		
		Binning	FKS	BAKS	Binning	FKS	BAKS
Number of units	$\{50, 75, \cdots, 200\}$	200	200	200	150	175	200
Number of epochs	$\{2, 3, \cdots, 8\}$	6	7	8	6	5	6
Batch size	$\{64, 96, 128\}$	64	96	64	64	32	32
Dropout rate	$\{0, 0.1, \cdots, 0.5\}$	0	0.2	0.2	0	0.5	0.1
Learning rate	$\{5, 10, \cdots, 50\} \times 10^{-4}$	0.002	0.004	0.005	0.0035	0.0035	0.003

where $\mathbf{x}, \mathbf{h}, \mathbf{f}, \mathbf{i}, \mathbf{o}, \mathbf{c}$ consecutively represent the input, output, forget gate, input gate, output gate, and memory cell. The operators $\odot$, σ, and $\tanh$ denote the element-wise multiplication, logistic sigmoid function, and hyperbolic tangent function, respectively. Matrices $\mathbf{W}, \mathbf{U}$ and bias vectors $\mathbf{b}$ represent the input and recurrent weights, respectively.

We empirically selected the number of layers and number of timesteps to be one and two, respectively. The last timestep from the LSTM output was connected to a fully connected layer to obtain the final output. Other hyperparameters, which include the number of units, number of epochs, batch size, dropout rate and learning rate, were determined through hyperparameter optimisation from predefined ranges (Table 1). The hyperparameter optimisation was conducted using a Bayesian optimisation package called Hyperopt [30] separately for each spike signal and firing rate estimation method and was run for 300 iterations. The resulting optimised hyperparameters are shown in Table 1. To save the computational time of experiments, the hyperparameter optimisation was performed only once using the first recording session; for the subsequent sessions, we used the same hyperparameter configuration. The LSTM decoders were implemented using Keras/TensorFlow deep learning framework [31] and trained using RMSprop optimiser with root mean squared error (RMSE) loss function.

2) Kalman filter (KF)

KFs have been employed in numerous BMI studies to predict hand kinematics or kinetics from neural signals [23], [32]–[35]. KF combines process and measurement models with the assumption that both models are linear and Gaussian. The process model defines the evolution of the state (assumed to be Markovian) from the previous timestep $t-1$ to the current timestep t as:

$$\mathbf{y}_t = \mathbf{F}\mathbf{y}_{t-1} + \mathbf{q}_t, \quad \mathbf{q}_t \sim \mathcal{N}(0, \mathbf{Q}) \tag{11}$$

where $\mathbf{F}$ denotes the state transition matrix that linearly maps the previous state to the current state; $\mathbf{q}_t$ represents the process noise. The state $\mathbf{y}_t$ at timestep t is formulated as:

$$\mathbf{y}_t = [s_x, s_y, v_x, v_y, a_x, a_y]_t^T \tag{12}$$

where $s_x, s_y, v_x, v_y, a_x, a_y$ denote the hand position, velocity, and acceleration in x and y directions, respectively; d refers to the dimension of the state. The measurement model describes the relationship between the state and the measurement at the current timestep t, expressed as:

$$\mathbf{x}_t = \mathbf{H}\mathbf{y}_t + \mathbf{r}_t, \quad \mathbf{r}_t \sim \mathcal{N}(0, \mathbf{R}) \tag{13}$$

where $\mathbf{H}$ denotes the measurement matrix that linearly transform the state $\mathbf{y}_t$ into the measurement $\mathbf{x}_t$; $\mathbf{r}_t$ represents the measurement noise.

KF is composed of two steps, *prediction* and *update*, which are performed in a recursive manner. In the prediction step, *a priori* state estimate ($\hat{\mathbf{y}}'_t$) and *a priori* error covariance ($\mathbf{P}'_t$) at the current timestep are predicted from previous state estimate ($\hat{\mathbf{y}}'_{t-1}$) and error covariance ($\mathbf{P}_{t-1}$),

$$\begin{aligned} \hat{\mathbf{y}}'_t &= \mathbf{F}\hat{\mathbf{y}}_{t-1} \\ \mathbf{P}'_t &= \mathbf{F}\mathbf{P}_{t-1}\mathbf{F}^T + \mathbf{Q} \end{aligned} \tag{14}$$

In the update step, *a posteriori* state estimate ($\hat{\mathbf{y}}_t$) and *a posteriori* error covariance ($\mathbf{P}_t$) at the current timestep are obtained from the predicted state estimate and error covariance combined with the information from the current measurement using the following equations:

$$\begin{aligned} \hat{\mathbf{y}}_t &= \hat{\mathbf{y}}'_t + \mathbf{K}_t(\mathbf{x}_t - \mathbf{H}\hat{\mathbf{y}}'_t) \\ \mathbf{P}_t &= (\mathbf{I} - \mathbf{K}_t\mathbf{H})\mathbf{P}'_t \end{aligned} \tag{15}$$

where $\mathbf{K}_t$ is the Kalman gain that represents how much weight given to the measurements to refine the current state estimate.

Following Wu *et al.*'s work [32], parameters $\mathbf{F}$, $\mathbf{H}$, $\mathbf{Q}$, and $\mathbf{R}$ were assumed to be invariant and were estimated using least square regression on the training data.

3) Wiener filter (WF)

WF linearly estimates kinematic data from the neural signals with the assumption of known stationary signal and additive noise. WF produces an optimal estimate in minimum mean squared error sense. WFs were employed in a number of prior BMI studies [3], [5], [6], [36], [37]. An estimate of hand kinematics, $\hat{y}(t)$, is expressed as:

$$\hat{y}(t) = \sum_{i=1}^{C} \sum_{\tau=0}^{L-1} w_i(\tau) x_i(t-\tau) \tag{16}$$

where $w_i(\tau)$ denotes a filter kernel and x_i represents the measurements (i.e. firing rates). C is the number of channels, whereas L is the number of lags/taps. L was empirically set to 15, and the filter weights ($w_i(\tau)$) were estimated using linear least-squares on the training data.

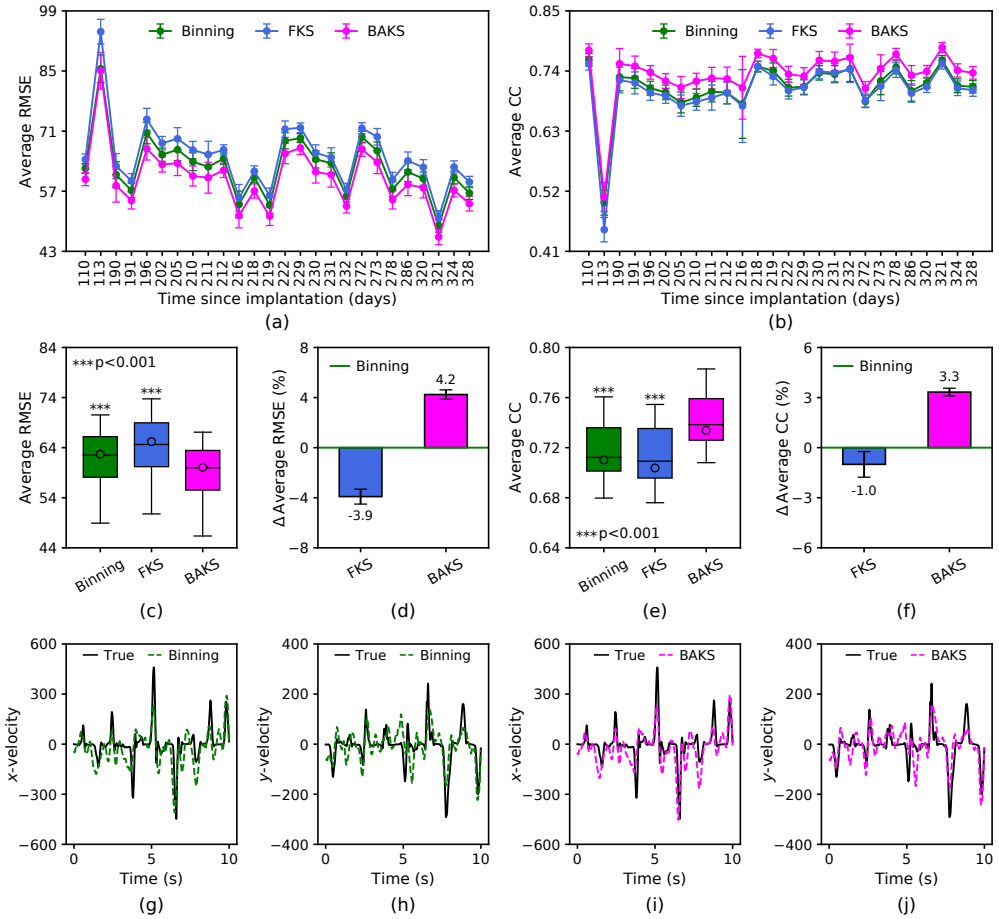

FIGURE 2. Comparison of decoding performance of MUA-driven KF decoder across different firing rate estimation methods. (a),(b) Performance comparison over long-term recording sessions measured in RMSE and CC, respectively. (c),(e) Boxplot comparison across sessions measured in RMSE and CC, respectively. Asterisks indicate firing rate estimation methods whose performances differed significantly from that of BAKS (*** p<0.001). (d),(f) Performance improvement/degradation (in percent RMSE and CC, respectively) relative to binning. Positive (negative) value indicates performance improvement (degradation). Black error bars denote 95% confidence intervals. (g)-(j) Snippet examples of true and decoded velocities in x- and y- coordinates from different firing rate estimation methods (data from recording session I20170131_02).

F. PERFORMANCE EVALUATION AND METRICS

The neural data were divided into 10 non-overlapping contiguous blocks of equal size which were categorised further into three sets: training (8 concatenated blocks), validation (1 block) and testing (1 block). The training, validation, and testing sets were used to train, optimise, and evaluate the decoder, respectively.

Decoding performance was evaluated using two commonly used metrics: (1) root mean square error (RMSE) and (2) Pearson's correlation coefficient (CC), which are formulated as follows:

$$\text{RMSE} = \sqrt{\sum_{i=1}^{N}(\hat{y}_i - y_i)^2/N} \tag{17}$$

$$\text{CC} = \frac{\sum_{t=1}^{N}(y_t - \bar{y})(\hat{y}_t - \bar{\hat{y}}_t)}{\sqrt{\sum_{t=1}^{N}(y_t - \bar{y})^2}\sqrt{\sum_{t=1}^{N}(\hat{y}_t - \bar{\hat{y}}_t)^2}} \tag{18}$$

where y_t and $\hat{y}_t$ denote the true and decoded hand kinematics at timestep t, respectively, and N represents the total number of samples.

For each session, the mean and confidence interval of the decoding performance were evaluated on 10 different blocks within the testing set. To test statistical significance between a pair of different decoders, a two-tailed paired t-test was used if the difference between the pairs follows normal distribution; otherwise, a two-tailed paired Wilcoxon signed-rank test was used. The significance level (α) was set to 0.05.

When using boxplot for visualisation, the horizontal line and circle mark inside each boxplot represent the median and mean, respectively. The coloured solid box represents interquartile range (from 25th to 75th percentiles). The whisker extends 1.5 times the interquartile range.

III. RESULTS

A. PERFORMANCE COMPARISON ACROSS FIRING RATE ESTIMATION METHODS

First, we evaluated the decoding performance of each firing rate estimation method using MUA-driven KF decoder. Figs. 2a and 2b present long-term decoding performance

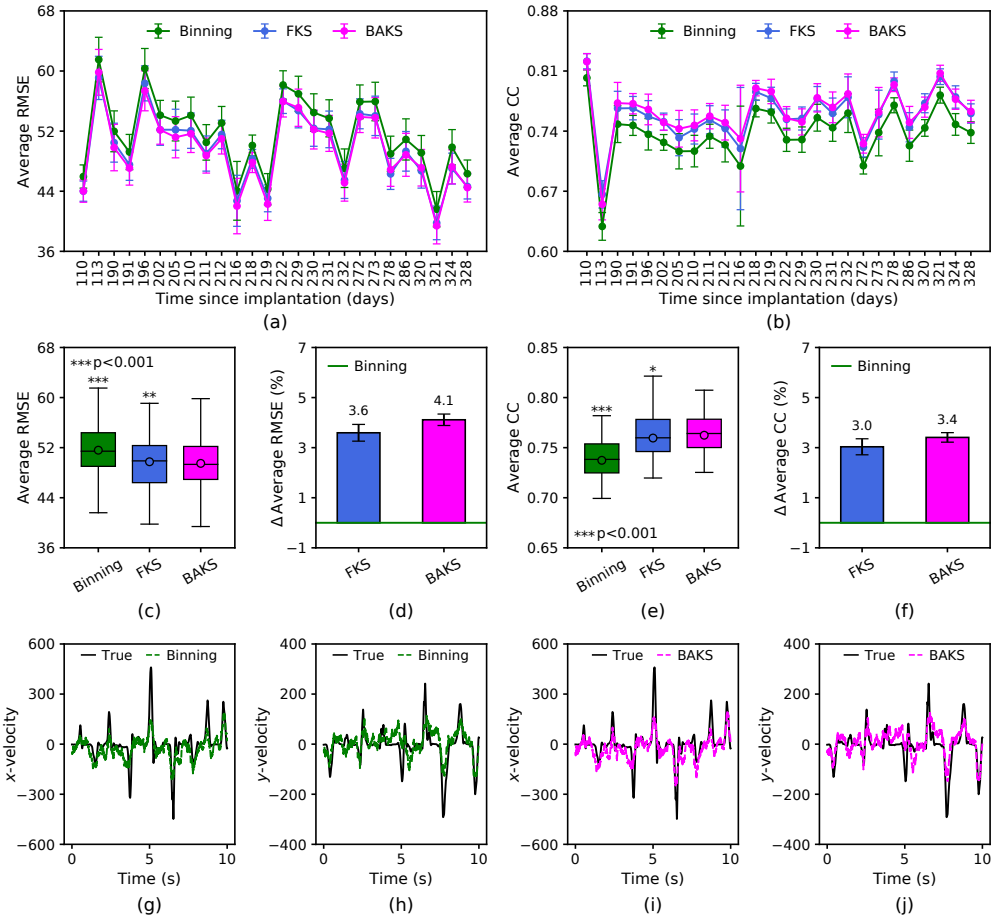

FIGURE 3. Comparison of decoding performance of MUA-driven WF decoder across different firing rate estimation methods. (a),(b) Performance comparison over long-term recording sessions measured in RMSE and CC, respectively. (c),(e) Boxplot comparison across sessions measured in RMSE and CC, respectively. Asterisks indicate firing rate estimation methods whose performances differed significantly from that of BAKS (* p<0.05, ** p<0.01, *** p<0.001). (d),(f) Performance improvement/degradation (in percent RMSE and CC, respectively) relative to binning. Positive (negative) value indicates performance improvement (degradation). Black error bars denote 95% confidence intervals. (g)-(j) Snippet examples of true and decoded velocities in x- and y- coordinates from different firing rate estimation methods (data from recording session I20170131_02).

comparison across firing rate estimations methods, measured in RMSE and CC, respectively. We found that BAKS outperformed both binning and FKS in all metrics across all recording sessions (100%). The average decoding performance of each method was as follows (sorted from highest to lowest): BAKS (RMSE = 60.08 ± 1.44, CC = 0.73 ± 0.01), binning (RMSE = 62.70 ± 1.40, CC = 0.71 ± 0.01), and FKS (RMSE = 65.21 ± 1.60, CC = 0.70 ± 0.01). The RMSE and CC values are written in terms of mean $\pm$ standard error of the mean (SEM). Compared to binning, BAKS yielded an average performance improvement of 4.2% (RMSE) and 3.3% (CC); however, FKS exhibited an average performance degradation of 3.9% (RMSE) and 1.0% (CC), as can be seen in Figs. 2d and 2f. Statistical tests showed that the performance of BAKS differed significantly from that of other methods (in both RMSE and CC metrics) as shown in Figs. 2c and 2e. Snippet examples of actual and decoded velocities (in x- and y-directions) from binning and BAKS are illustrated in Figs. 2g-j.

To determine whether the above findings were also observed when using different decoding algorithms, we performed decoding comparison across firing rate estimation methods using WF and LSTM decoders. We found that BAKS consistently outperformed other methods as shown in Fig. 3 for WF decoder and Fig. 4 for LSTM decoder. There was a statistical significant difference in decoding performance between BAKS and other methods as illustrated in Figs. 3c,e (WF decoder) and Figs. 4c,e (LSTM decoder). In the case of WF decoder, according the decoding performance, we found the following descending order: BAKS (RMSE = 49.49 ± 0.98, CC = 0.76 ± 0.01) > FKS (RMSE = 49.75 ± 0.96, CC = 0.76 ± 0.01) > binning (RMSE = 51.60 ± 0.99, CC = 0.74 ± 0.01). Relative to binning, BAKS achieved 4.11% (3.41%) average performance improvement in RMSE (CC), whereas FKS gained 3.59% (3.03%) average performance improvement in RMSE (CC), as shown in Fig. 3d (3f). In the case of LSTM decoder, we obtained the following descending order: BAKS (RMSE = 41.56 ± 0.84,

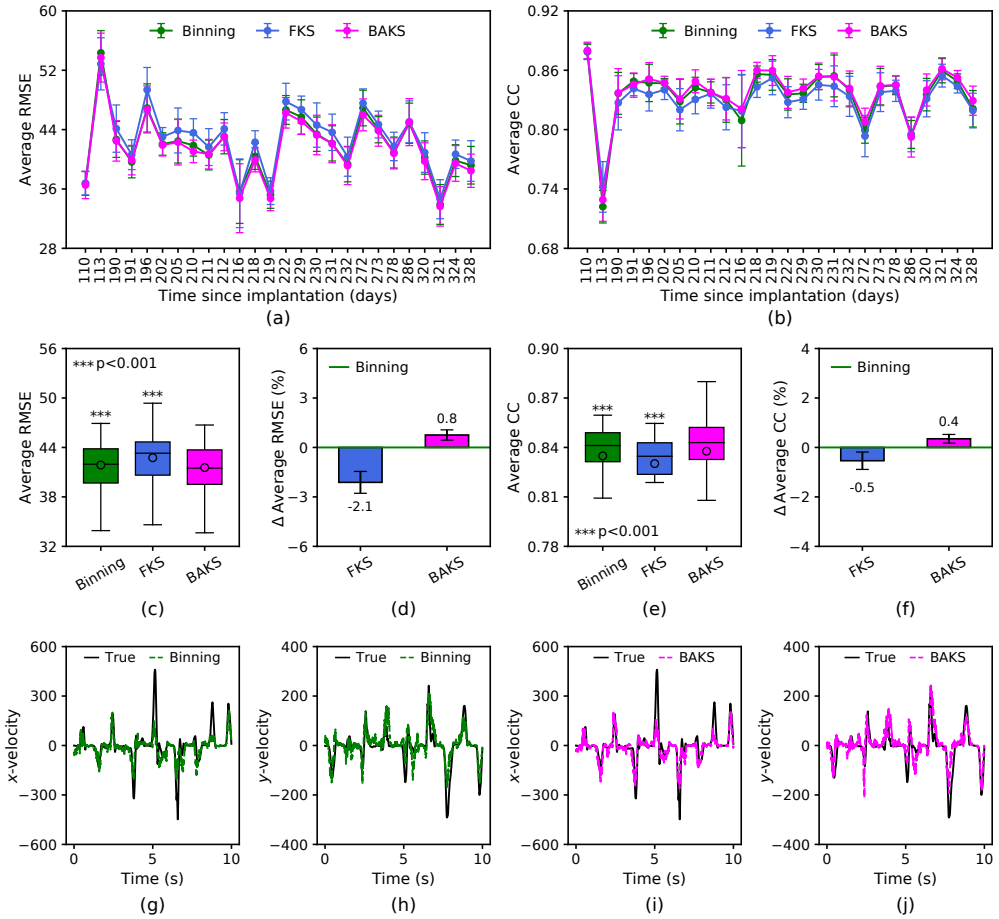

FIGURE 4. Comparison of decoding performance of MUA-driven LSTM decoder across different firing rate estimation methods. (a),(b) Performance comparison over long-term recording sessions measured in RMSE and CC, respectively. (c),(e) Boxplot comparison across sessions measured in RMSE and CC, respectively. Asterisks indicate firing rate estimation methods whose performances differed significantly from that of BAKS (*** p<0.001). (d),(f) Performance improvement/degradation (in percent RMSE and CC, respectively) relative to binning. Positive (negative) value indicates performance improvement (degradation). Black error bars denote 95% confidence intervals. (g)-(j) Snippet examples of true and decoded velocities in x- and y- coordinates from different firing rate estimation methods (data from recording session I20170131_02).

CC = 0.84 ± 0.01) > binning (RMSE = 41.87 ± 0.85, CC = 0.83 ± 0.01) > FKS (RMSE = 42.75 ± 0.84, CC = 0.83 ± 0.00). The average performance improvement achieved by BAKS relative to binning in the case of LSTM was considerably smaller (RMSE = 0.76%, CC = 0.35%) than in the cases of KF and WF (see Figs. 4d,f). Examples of actual and decoded velocities (in x- and y-directions) in the cases of WF and LSTM are plotted in Figs. 3g-j and Figs. 4g-j, respectively.

B. IMPACT OF WINDOW WIDTH ON DECODING PERFORMANCE

Next, we assessed the impact of firing rate estimation methods under different window widths on decoding performance. For each firing rate estimation method, we varied the value of window width from 16 ms to 400 ms with an increment of 16 ms. We used KF and LSTM decoders for performance comparison. Figs. 5a,b and 5c,d illustrate the impact of varying window widths on decoding performance using KF and LSTM decoders, respectively. Results showed that increasing the window width up to a certain value would improve the decoding performance; above this value, however, the decoding performance reached a plateau or tended to decrease. Binning and BAKS were found to reach the plateau level faster than FKS in both KF and LSTM decoders. All firing rate estimation methods reached the plateau level faster in the case of LSTM decoder than in the case of KF decoder.

C. PERFORMANCE COMPARISON ACROSS DECODING ALGORITHMS

Using BAKS as the firing rate estimation method, we then evaluated the decoding performance of different decoders (KF, WF, and LSTM). Figs. 6a and 6b present the decoding performance comparison over 26 sessions in terms of RMSE and CC, respectively. We found that LSTM outperformed other decoders in all sessions. By sorting descendingly according to the decoding performance, we obtained the following order: LSTM (RMSE = 41.56 ± 0.84, CC = 0.84 ± 0.01), WF (RMSE = 49.49 ± 0.98, CC = 0.76 ± 0.01),

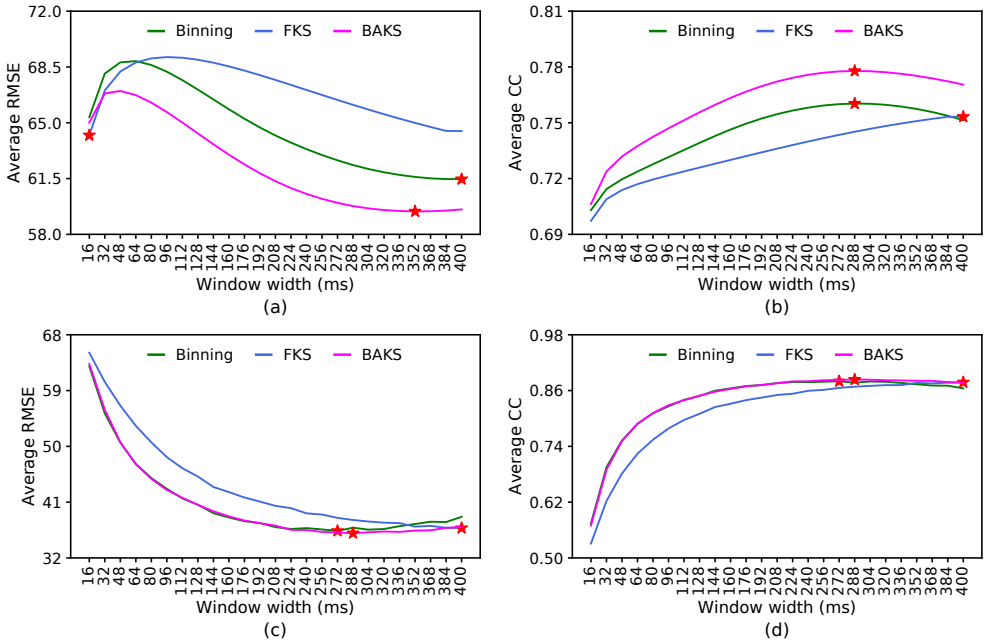

FIGURE 5. Decoding performance comparison across different firing rate estimation methods with varying window widths. (a),(b) Performance comparison using MUA-driven KF decoder measured in RMSE and CC, respectively. (c),(d) Performance comparison using MUA-driven LSTM decoder measured in RMSE and CC, respectively. Red stars indicate window widths that yielded the best decoding performances on validation set from session I20160627_01.

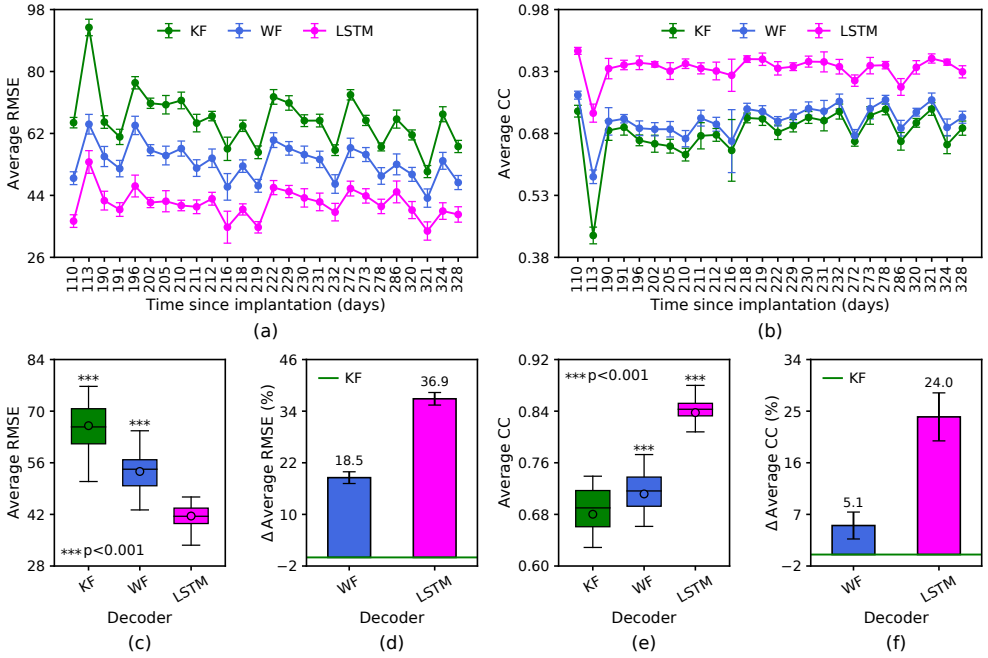

FIGURE 6. Comparison of decoding performance of MUA-driven decoders with firing rates computed by BAKS. (a),(b) Performance comparison over long-term recording sessions measured in RMSE and CC, respectively. (c),(e) Boxplot comparison across sessions measured in RMSE and CC, respectively. Asterisks indicate decoders that yielded statistically significant different performances from that of binning (*** p<0.001). (d),(f) Performance improvement/degradation (in percent RMSE and CC, respectively) relative to binning. Positive (negative) value indicates performance improvement (degradation). Black error bars denote 95% confidence intervals.

and KF (RMSE = 60.08 ± 1.44, CC = 0.73 ± 0.01) as observed from Figs. 6c and 6e. Relative to KF, LSTM yielded an average performance improvement of 30.64% (RMSE) and 14.56% (CC), whereas WF yielded an average performance improvement of 17.41% (RMSE) and 4.20% (CC) as shown in Figs. 6d and 6f. Statistical tests showed that there were

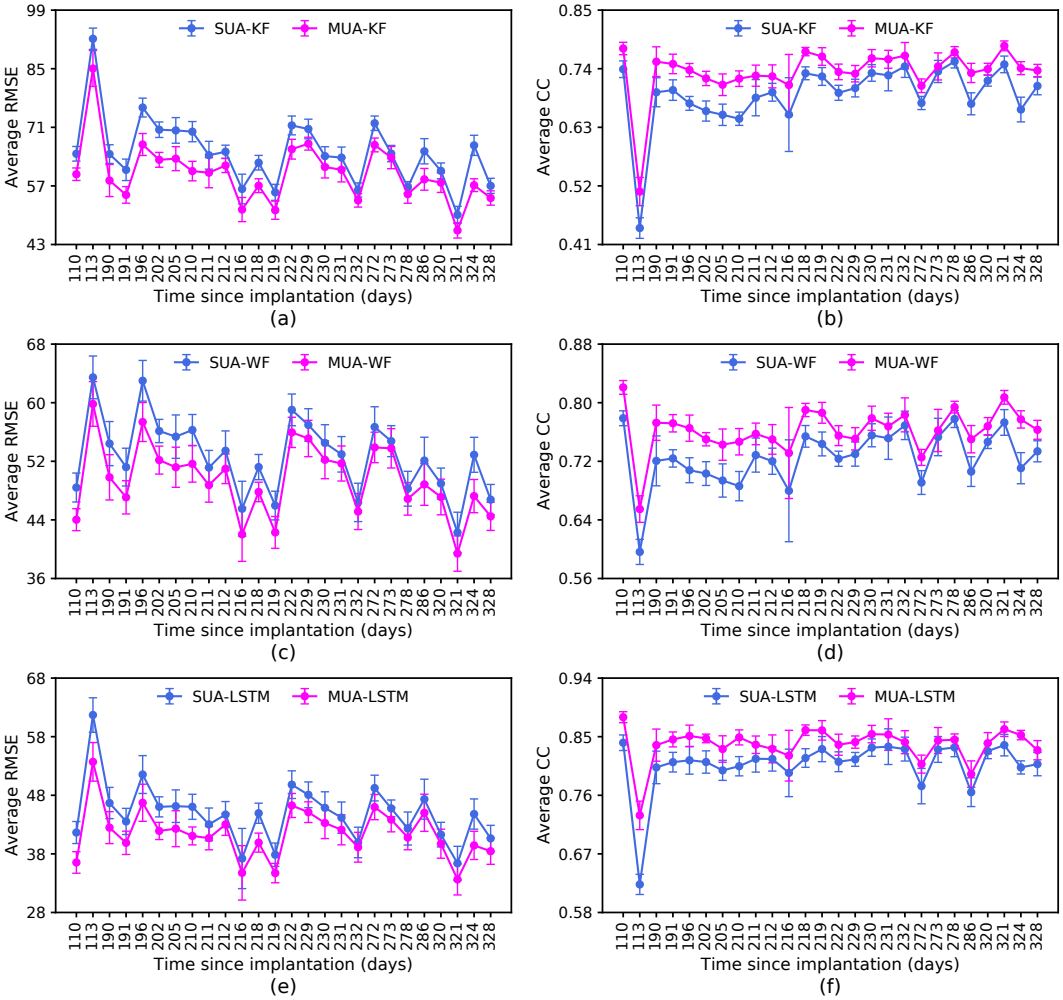

FIGURE 7. Comparison of decoding performance between SUA- and MUA-driven decoders with firing rates computed by BAKS. (a),(b) Long-term decoding comparison between SUA- and MUA-driven KF decoders measured in RMSE and CC, respectively. (c),(d) Long-term decoding comparison between SUA- and MUA-driven WF decoders measured in RMSE and CC, respectively. (e),(f) Long-term decoding comparison between SUA- and MUA-driven LSTM decoders measured in RMSE and CC, respectively.

statistically significant differences in decoding performance between LSTM and other decoders ($p<0.001$).

D. DECODING PERFORMANCE COMPARISON BETWEEN SUA AND MUA

Lastly, we compared the decoding performance of MUA against SUA using BAKS coupled with different decoders. Figs. 7a-b, Figs. 7c-d, and Figs. 7e-f plot the decoding performance comparison between MUA and SUA across 26 sessions for the case of KF, WF, and LSTM decoders, respectively. Extensive results revealed that MUA yielded significantly better decoding performance than SUA in all sessions regardless of the decoders. Relative to SUA, average performance improvement achieved by MUA from each decoding algorithm was as follows: KF (RMSE = 7.53%, CC = 6.34%), WF (RMSE = 5.94%, CC = 5.21%), and LSTM (RMSE = 7.26%, CC = 3.94%).

IV. DISCUSSION

We aim to improve the decoding performance of MUA-based BMI by proposing BAKS as a feature extraction method coupled with LSTM as a decoding algorithm. Firstly, we compared the decoding performance of BAKS against two commonly used feature extractions methods (binning and FKS) under three different decoding algorithms (KF, WF, and LSTM). Comparison results demonstrated that BAKS consistently outperformed other methods across different decoding algorithms. When varying the window width values, we also observed consistently better performance of BAKS compared to other methods. This could be attributed to BAKS' capability to obtain a smoother and more accurate estimate of firing rate. BAKS incorporates a data-driven and adaptive bandwidth parameter that allows for more accurate estimation of firing rate when there is a rapidly changing spike dynamic. On the other hand, both binning and FKS employ a fixed, predefined bandwidth parameter; thus, they

cannot accurately estimate the firing rate from a spike train with rapidly changing spike dynamic.

We found that the average performance improvement of BAKS relative to other methods was more significant in the case of linear decoders (KF and WF) than deep learning decoder (LSTM). This might be because LSTM can compensate for the differences in estimated firing rates through gradient-based optimisation algorithm during the training. In other words, LSTM is less sensitive to the smoothness and accuracy of estimated firing rates than both KF and WF. Using BAKS as the feature extraction algorithm, we then compared the performance of LSTM against KF and WF. Results showed that LSTM significantly outperformed both KF and WF, which demonstrates the effectiveness of LSTM in capturing the complex, non-linear relationship between neural signals and hand kinematic data.

Lastly, using BAKS coupled with LSTM, we compared the decoding performance of MUA to that of SUA. Empirical results revealed that MUA achieved significantly higher decoding performance than SUA. When using BAKS coupled with KF or WF, we also observed the same finding. These results contradict several prior studies where SUA was shown to yield better decoding performance than MUA [11], [14], [16], [38]. It is difficult to find the exact reason to this contradiction due to the differences in recording setup, behavioural task, signal processing, decoding algorithm, etc. across studies. One possible explanation is that in our study, to obtain SUA, we only used well-isolated (sorted) spikes and discarded unsorted spikes (also known 'noise' or 'hash' units). Hash units contained all spikes that did not match any of the operator's defined templates used for spike sorting. Todorova *et al.* have recently shown that hash units contained some information about movement and discarding this information could degrade the decoding performance [24]. On the contrary, when computing MUA, we used all the detected spikes, including the hash units, which potentially contributed to improved decoding performance.

This present study expands our previous conference paper [39] by adding the following contributions: (1) proposing MUA as an alternative input signal and comparing its decoding performance to that of SUA, (2) proposing deep learning decoder and comparing its decoding performance to that of linear decoder, (3) using chronic neural data spanning more than 7.3 months of recording sessions, and (4) adding FKS and WF for performance benchmark. In summary, overall results demonstrated that our proposed methods, BAKS and LSTM, achieved better decoding performance than other methods.

V. CONCLUSION

We have presented BAKS and LSTM for MUA-based BMI and evaluated its decoding performance on chronic neural recordings. We have shown that BAKS coupled with LSTM significantly outperformed combinations of other firing rate estimation methods and decoding algorithms. This suggests the feasibility and the potential use of BAKS and LSTM for improving the decoding performance of MUA-based BMIs.

ACKNOWLEDGMENT

We thank J. E. O'Doherty and P. N. Sabes for making their data publicly available.

REFERENCES

[1] M. D. Serruya, N. G. Hatsopoulos, L. Paninski, M. R. Fellows, and J. P. Donoghue, "Instant neural control of a movement signal," Nature, vol. 416, no. 6877, pp. 141–142, 2002.

[2] D. M. Taylor, S. I. H. Tillery, and A. B. Schwartz, "Direct cortical control of 3D neuroprosthetic devices," Science, vol. 296, no. 5574, pp. 1829–1832, 2002.

[3] 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. Nicolelis, "Learning to control a brain–machine interface for reaching and grasping by primates," PLOS Biol., vol. 1, no. 2, p. e42, 2003.

[4] F. R. Willett, A. J. Suminski, A. H. Fagg, and N. G. Hatsopoulos, "Improving brain–machine interface performance by decoding intended future movements," J. Neural Eng., vol. 10, no. 2, p. 026011, 2013.

[5] L. R. Hochberg, M. D. Serruya, G. M. Friehs, J. A. Mukand, M. Saleh, A. H. Caplan, A. Branner, D. Chen, R. D. Penn, and J. P. Donoghue, "Neuronal ensemble control of prosthetic devices by a human with tetraplegia," Nature, vol. 442, no. 7099, p. 164, 2006.

[6] S.-P. Kim, J. D. Simeral, L. R. Hochberg, J. P. Donoghue, and M. J. Black, "Neural control of computer cursor velocity by decoding motor cortical spiking activity in humans with tetraplegia," J. Neural Eng., vol. 5, no. 4, p. 455, 2008.

[7] J. L. Collinger, B. Wodlinger, J. E. Downey, W. Wang, E. C. Tyler-Kabara, D. J. Weber, A. J. McMorland, M. Velliste, M. L. Boninger, and A. B. Schwartz, "High-performance neuroprosthetic control by an individual with tetraplegia," The Lancet, vol. 381, no. 9866, pp. 557–564, 2013.

[8] B. Wodlinger, J. Downey, E. Tyler-Kabara, A. Schwartz, M. Boninger, and J. Collinger, "Ten-dimensional anthropomorphic arm control in a human brain- machine interface: difficulties, solutions, and limitations," J. Neural Eng., vol. 12, no. 1, p. 016011, 2014.

[9] S. Suner, M. R. Fellows, C. Vargas-Irwin, G. K. Nakata, and J. P. Donoghue, "Reliability of signals from a chronically implanted, silicon-based electrode array in non-human primate primary motor cortex," IEEE Trans. Neural Syst. Rehabil. Eng, vol. 13, no. 4, pp. 524–541, 2005.

[10] J. C. Barrese, N. Rao, K. Paroo, C. Triebwasser, C. Vargas-Irwin, L. Franquemont, and J. P. Donoghue, "Failure mode analysis of silicon-based intracortical microelectrode arrays in non-human primates," J. Neural Eng., vol. 10, no. 6, p. 066014, 2013.

[11] D. Wang, Q. Zhang, Y. Li, Y. Wang, J. Zhu, S. Zhang, and X. Zheng, "Long-term decoding stability of local field potentials from silicon arrays in primate motor cortex during a 2D center out task," J. Neural Eng., vol. 11, no. 3, p. 036009, 2014.

[12] M. D. Serruya, "Bottlenecks to clinical translation of direct brain-computer interfaces," Frontiers in systems neuroscience, vol. 8, p. 226, 2014.

[13] M. D. Murphy, D. J. Guggenmos, D. T. Bundy, and R. J. Nudo, "Current challenges facing the translation of brain computer interfaces from preclinical trials to use in human patients," Front. Cell. Neurosci., vol. 9, p. 497, 2016.

[14] G. W. Fraser, S. M. Chase, A. Whitford, and A. B. Schwartz, "Control of a brain–computer interface without spike sorting," J. Neural Eng., vol. 6, no. 5, p. 055004, 2009.

[15] C. A. Chestek, V. Gilja, P. Nuyujukian, J. D. Foster, J. M. Fan, M. T. Kaufman, M. M. Churchland, Z. Rivera-Alvidrez, J. P. Cunningham, S. I. Ryu et al., "Long-term stability of neural prosthetic control signals from silicon cortical arrays in rhesus macaque motor cortex," J. Neural Eng., vol. 8, no. 4, p. 045005, 2011.

[16] B. P. Christie, D. M. Tat, Z. T. Irwin, V. Gilja, P. Nuyujukian, J. D. Foster, S. I. Ryu, K. V. Shenoy, D. E. Thompson, and C. A. Chestek, "Comparison of spike sorting and thresholding of voltage waveforms for intracortical brain–machine interface performance," J. Neural Eng., vol. 12, no. 1, p. 016009, 2014.

[17] M. M. Shanechi, "Brain–machine interface control algorithms," IEEE Trans. Neural Syst. Rehabil. Eng., vol. 25, no. 10, pp. 1725–1734, 2017.

[18] A. J. Brockmeier and J. C. Príncipe, "Decoding algorithms for brain–machine interfaces," in Neural engineering. Springer, 2013, pp. 223–257.

[19] J. C. Kao, S. D. Stavisky, D. Sussillo, P. Nuyujukian, and K. V. Shenoy, "Information systems opportunities in brain–machine interface decoders," Proc. IEEE, vol. 102, no. 5, pp. 666–682, 2014.
[20] D. L. Menzer, H. Bokil, J. W. Ryou, N. D. Schiff, K. P. Purpura, and P. P. Mitra, "Characterization of trial-to-trial fluctuations in local field potentials recorded in cerebral cortex of awake behaving macaque," J. Neurosci. Methods, vol. 186, no. 2, pp. 250–261, 2010.
[21] N. Ahmadi, T. G. Constandinou, and C.-S. Bouganis, "Estimation of neuronal firing rate using Bayesian adaptive kernel smoother (BAKS)," PLoS One, vol. 13, no. 11, 2018.
[22] J. P. Cunningham, V. Gilja, S. I. Ryu, and K. V. Shenoy, "Methods for estimating neural firing rates, and their application to brain–machine interfaces," Neural Networks, vol. 22, no. 9, pp. 1235–1246, 2009.
[23] L. R. Hochberg, D. Bacher, B. Jarosiewicz, N. Y. Masse, J. D. Simeral, J. Vogel, S. Haddadin, J. Liu, S. S. Cash, P. van der Smagt et al., "Reach and grasp by people with tetraplegia using a neurally controlled robotic arm," Nature, vol. 485, no. 7398, p. 372, 2012.
[24] S. Todorova, P. Sadtler, A. Batista, S. Chase, and V. Ventura, "To sort or not to sort: the impact of spike-sorting on neural decoding performance," J. Neural Eng., vol. 11, no. 5, p. 056005, 2014.
[25] J. E. O'doherty, M. M. B. Cardoso, J. G. Makin, and P. N. Sabes, "Nonhuman primate reaching with multichannel sensorimotor cortex electrophysiology," Zenodo http://doi.org/10.5281/zenodo.583331, 2017.
[26] J. G. Makin, J. E. O'Doherty, M. M. Cardoso, and P. N. Sabes, "Superior arm-movement decoding from cortex with a new, unsupervised-learning algorithm," J. Neural Eng., vol. 15, no. 2, p. 026010, 2018.
[27] N. Ahmadi, T. G. Constandinou, and C.-S. Bouganis, "Decoding hand kinematics from local field potentials using long short-term memory (LSTM) network," in IEEE/EMBS Conf. Neural Eng. (NER). IEEE, 2019, pp. 415–419.
[28] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.
[29] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," IEEE Trans. Neural Syst. Rehabil. Eng, vol. 28, no. 10, pp. 2222–2232, 2017.
[30] J. Bergstra, B. Komer, C. Eliasmith, D. Yamins, and D. D. Cox, "Hyperopt: a python library for model selection and hyperparameter optimization," Comput. Sci. Discovery, vol. 8, no. 1, p. 014008, 2015.
[31] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard et al., "TensorFlow: A system for large-scale machine learning," in Proc. USENIX Symp. OS Design Impl. (OSDI), 2016, pp. 265–283.
[32] W. Wu, M. J. Black, Y. Gao, M. Serruya, A. Shaikhouni, J. Donoghue, and E. Bienenstock, "Neural decoding of cursor motion using a kalman filter," in Adv. Neural Inf. Process. Syst., 2003, pp. 133–140.
[33] W. Wu, Y. Gao, E. Bienenstock, J. P. Donoghue, and M. J. Black, "Bayesian population decoding of motor cortical activity using a kalman filter," Neural Comput., vol. 18, no. 1, pp. 80–118, 2006.
[34] G. H. Mulliken, S. Musallam, and R. A. Andersen, "Decoding trajectories from posterior parietal cortex ensembles," J. Neurosci., vol. 28, no. 48, pp. 12 913–12 926, 2008.
[35] W. Wu and N. G. Hatsopoulos, "Real-time decoding of nonstationary neural activity in motor cortex," IEEE Trans. Neural Syst. Rehabil. Eng., vol. 16, no. 3, pp. 213–222, 2008.
[36] N. Hatsopoulos, J. Joshi, and J. G. O'Leary, "Decoding continuous and discrete motor behaviors using motor and premotor cortical ensembles," J. Neurophysiol., vol. 92, no. 2, pp. 1165–1174, 2004.
[37] S.-P. Kim, J. C. Sanchez, Y. N. Rao, D. Erdogmus, J. M. Carmena, M. A. Lebedev, M. Nicolelis, and J. Principe, "A comparison of optimal MIMO linear and nonlinear models for brain–machine interfaces," J. Neural Eng., vol. 3, no. 2, p. 145, 2006.
[38] S. Perel, P. T. Sadtler, E. R. Oby, S. I. Ryu, E. C. Tyler-Kabara, A. P. Batista, and S. M. Chase, "Single-unit activity, threshold crossings, and local field potentials in motor cortex differentially encode reach kinematics," J. Neurophysiol., vol. 114, no. 3, pp. 1500–1512, 2015.
[39] N. Ahmadi, T. G. Constandinou, and C.-S. Bouganis, "Spike rate estimation using Bayesian adaptive kernel smoother (BAKS) and its application to brain-machine interfaces," in 40th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC). IEEE, 2018, pp. 2547–2550.