A Co-Design of Hardware and Software Systems for Future Integrated Brain-Computer Interfaces

Rajeev B. Botadra[1], Hao Fang[1], Chi-Jui Chen[2], Leo Scholl[1], Yan-Lun Huang[4], Trung Le[1], Ryan Canfield[5], Shih-Chieh Hsu[7], Eli Shlizerman[6], Bo-Cheng Lai[2], Amy Orsborn[1,3,5], Scott Hauck[1]

***Abstract*— Rapid advances in systems neuroscience and machine learning have greatly expanded the ability to build brain-computer interfaces (BCIs) for the investigation of neural mechanisms in brain functions and the treatment of neural disorders. Currently, BCI technologies are limited by hardware that cannot realize high-performance and low-latency brain decoding with high-density neural recordings for online inference tasks. Here, we propose a co-design of the hardware and software systems with hardware acceleration of state-of-the-art BCI technologies to address the above challenges. We first use the Neuropixel 1.0 system for high-density neural signal acquisition. Second, we leverage the well-established LFADS model to learn robust neural latent representations. Third, we quantize the LFADS model and deploy it on a Field Programmable Gate Array (FPGA) to achieve a sub-millisecond inference latency for real-time signal decoding. Last, we complete the loop using optogenetic stimulation in the brain. Our co-designed system can be treated as a closed-loop testbed, that can orchestrate real-time experiments to shed light on the workings of brain functions, investigating a promising direction for hardware acceleration in future BCIs.**

***Index Terms*— Brain-Computer Interfaces, FPGA, Hardware Acceleration, Neuropixel, Neural Decoding**

I. INTRODUCTION

In recent years, innovations in system neuroscience and machine learning have witnessed great advances in developing brain-computer interfaces (BCIs). Modern BCIs can be treated as engineering systems that build direct communications between the brain and external devices [1]. The current BCI systems include two main categories: the open-loop (unidirectional) BCI and the close-loop (bidirectional) BCI. The open-loop BCI mainly focuses on effective brain state decoding such as motor intentions [2] or mood state [3]. Later, the decoded brain state from brain activity can be applied to control external devices such as a simple computer cursor or complicated robotic arms to improve the control performance [2]. Beyond the brain state decoding application, the closed-loop BCI aims to close the loop via stimulating the brain to induce or inhibit activity [4]. For example, researchers have developed bidirectional BCI-based neuroprostheses [5] to restore brain dysfunction and communication between damaged brain regions, while others investigated closed-loop neuromodulation systems to stimulate the brain region for treating brain dysfunctions [6].

However, despite the active research in the BCI field, current studies mainly rely on computer software, i.e., machine learning approach for open-loop brain decoding [7]–[9] or Monte Carlo simulations for closed-loop study [10]. For example, by using the offline data collected from the monkey's behavior task, researchers can develop offline machine learning algorithms such as Latent Factor Analysis through Dynamical Systems (LFADS) to precisely decode the movement trajectories [8]. Also, using computational biophysical models, e.g., Parkinson's disease (PD) model in rats, researchers can study the closed-loop deep brain stimulation systems for the therapeutic performance of PD [11]. Recently, a few studies have stepped from purely computer investigation to real-time online evaluation. For example, a recent intracortical BCI study has illustrated high finger behavior decoding and control accuracy in real time [12]. Subjects with major depressive disorders stimulated via responsive neuromodulation systems show greater improvement in their mood state over time [13]. However, building a generalizable real-time BCI testbed remains challenging due to the integrated communication between the hardware and the software systems. Specifically, the integrated system has difficulties with the implanted neural recording device, the in-chip neural signal processing unit design, and the system communication and computational latency. To this end, building an integrated BCI system is of critical importance.

In our work, we propose a co-design of the hardware and software systems for integrated BCI applications. The main innovation is to build a system testbed by integrating state-of-the-art BCI technologies. Our system includes four essential components. First, we use Neuropixel 1.0 as the high-density neural signal acquisition system to record the real-time spiking activity from the brain. Second, we train a robust LFADS model using offline data. Third, we quantize the well-trained LFADS model and deploy it on Field Programmable Gate Arrays (FPGAs) to enable high inference speed with negligible system latency. Last, we propose to close the loop using optical stimulations. Overall, our BCI system achieves high decoding results while maintaining lower computational costs. Our design can be applied to many BCI studies such as movement decoding and behavior manipulation, which has the potential to shed light on the understanding of brain functions/dysfunctions and implications for future closed-loop BCI design.

[1]Department of Electrical and Computer Engineering at the University of Washington, Seattle WA, USA (email: hauck@uw.edu)

[2]College of Electrical and Computer Engineering, National Yang Ming Chiao Tung University, Hsin-Chu, Taiwan

[3]Washington National Primate Research Center, Seattle, WA, USA

[4]Department of Electrical and Computer Engineering at the University of Texas at Austin, Austin TX, USA

[5]Department of Bioengineering at the University of Washington, Seattle WA, USA

[6]Department of Applied Mathematics at the University of Washington, Seattle WA, USA

[7]Department of Physics at the University of Washington, Seattle WA, USA

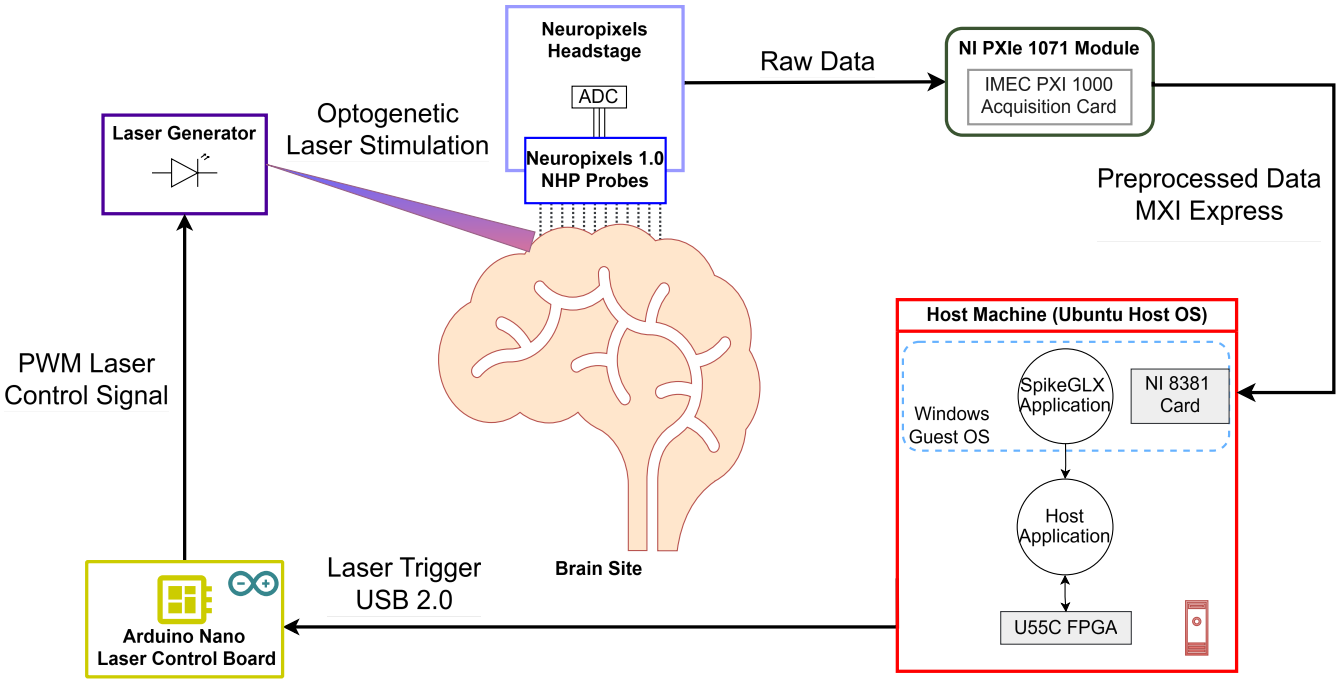

Fig. 1. The proposed co-design of hardware and software system for closed-loop BCIs. The neural activity of the brain site is recorded via Neuropixel probes and converted to a digital signal at the headstage. Next, the raw digital signal is sent to an intermediary data acquisition module for data pre-processing. The pre-processed data is sent to a host computer that runs a neural decoding algorithm accelerated by an FPGA, producing a decoded brain state. Then, the host sends a laser trigger signal to the laser control board. The control board converts the trigger to a Pulse-Width Modulated (PWM) signal to control the laser generator. Lastly, the laser generator stimulates the light-sensitive neurons in specific brain sites.

II. METHODS

A. System Components

The proposed co-designed system is functionally described by three subsystems: data acquisition using Neuropixel probes, FPGA accelerated neural signal processing, and optogenetic stimulation, forming a closed loop interface with the brain while maintaining a low system action latency (see Figure 1).

1) Data Acquisition System

The data acquisition system uses IMEC Neuropixel 1.0 NHP probes to capture high-density intracortical neural spiking activity. The probes offer a high spatial resolution of the recording site with 960 electrode tiles tightly placed along the shank, enabling localized studies of neural circuits. Each probe reads data from 384 channels (with a mapping to specific electrode tiles defined in configuration files) at a sampling frequency of 30 KHz. Thus each channel provides a high temporal resolution of the signal, sampling roughly every 333.3μs. Next, the raw neural signals are transmitted to the headstage which amplifies, digitizes, and serializes the data. The data is then sent to an IMEC PXIe Acquisition module housed in a NI PXIe 1071 chassis. After deserializing and formatting the signal (and offering additional functionality such as synchronizing signals from multiple probes simultaneously), the module sends the data to a host machine over a Gen2 x8 MXI Express cable. The host uses the SpikeGLX API to configure the recording parameters and control the data, batching incoming data into FIFO buffers for decoding. The buffers are then reshaped into 2D tiles ($C \times T$), where C and T represent the number of channels and time bins respectively. The host then transfers the data from RAM to a global memory buffer on the FPGA over a PCIe Gen3 x16 bus and invokes the LFADS decoding kernel.

2) Neural Signal Processing on FPGA

The FPGA at the core of the testbed is a Xilinx Alveo U55C, offering a balance between hardware resources to support the execution of larger decoding algorithms and cost. The U55C is not a standalone System-on-Chip (SoC) with an integrated processor; instead, it relies on a host machine to handle input and output data, allocate memory buffers, and invoke application kernels. Applications deployed on the FPGA are first written in High-Level Synthesis (HLS) code, synthesized to RTL, packaged into an IP core with required interfaces, and compiled into a kernel. This kernel is then launched on the FPGA by the host machine using Xilinx RunTime (XRT) drivers. Data (i.e. decoder model inputs, outputs) is passed to and from the FPGA over PCIe using system calls through XRT.

Typically, HLS designs are defined in C++ and synthesized into RTL. However, most ML-based decoders are developed in Python to leverage well-equipped libraries such as Tensorflow. To create a decoder-agnostic system and rapidly test, optimize, and deploy new ML models we use a novel package called HLS4ML: a tool that converts ML models defined in Python frameworks into latency and throughput-optimized HLS C++ designs [14].

The LFADS model was defined, trained, and quantized using Tensorflow and QKeras in Python. The quantized model was then converted into HLS using HLS4ML, and then compiled into an optimized kernel using the Vitis HLS pipeline. For inference, the model was provided a context window

of $T = 20ms$ (as standard in previous BCI literature [8]). However, waiting $20ms$ to generate each input batch would severely underutilize the FPGA's sub-millisecond inference latency. Instead, to leverage the available compute we utilize a sliding window realized through a FIFO buffer, pushing $\Delta = 1ms$ of data into the buffer to produce each new input batch. Once the host transfers a new batch to the kernel, it is forward propagated through the LFADS kernel to create a learned latent representation of the brain state. These latent factors are returned to the host machine and can be used in experiment-specific downstream tasks such as hand-movement prediction [8].

3) Optical Stimulation to Close the Loop

The host machine uses the decoded brain state generated by LFADS to stimulate the brain using optogenetics. The BCI testbed is equipped with a laser generator controlled by an Arduino Nano Laser Control Board. The board receives control signals from the host over a USB 2.0 connection and generates a pulse-width modulation (PWM) signal to control the laser. The specific stimulation pattern is directly encoded in the signal, with a binary 1/0 turning the laser ON/OFF (the duty cycle is set to 100% during the window of stimulation and 0% otherwise). The laser is directed toward the brain site with fiber optic cables, providing coarse spatial control of the location of stimulation. Still, the Arduino provides a fine (sub-microsecond) degree of control on the time of stimulation with a maximum PWM Switching Frequency of 4MHz.

The primary focus in evaluating our testbed was closed-loop performance and system action latency (i.e. the time between an input signal from the brain to stimulation). To this purpose, we implemented a simple stimulation encoding by triggering the laser when the average inferred latent factors exceeded a threshold. While the study of optical stimulation techniques for unknown neural foundations is beyond the scope of our co-design work, we provide several insights into future studies and directions (see *Discussion*). For instance, a study aimed at identifying the behavioral impacts of a particular motor circuit might trigger an inhibitory laser stimulation when the decoder predicts that the circuit will become active, thereby isolating the circuit to evaluate changes in behavior from baseline comparison (no optical stimulation). To this end, our testbed provides a hardware-accelerated platform capable of investigating more interesting neuroscience questions with different decoding and stimulation methodologies.

B. LFADS as Neural Decoder Model

We use the well-established LFADS architecture [8] to construct a latent neural representation of the brain state from neural spiking activity. A standard forward propagation through LFADS is as follows:

Encoder

$$X \in \mathbb{R}^{C\times T} \quad x_t \in \mathbb{R}^{C\times 1} \quad t = \{1,\dots,T\} \tag{1}$$

$$e_t^b = EncoderRNN(e_{t+1}^b, x_t) \tag{2}$$

$$e_t^f = EncoderRNN(e_{t-1}^f, x_t) \tag{3}$$

$$h = [e_1^b, e_T^f] \tag{4}$$

Decoder

$$\mu = W^{\mu}\cdot h \tag{5}$$

$$\sigma = \exp(\frac{1}{2}W^{\sigma}\cdot h) \tag{6}$$

$$\hat{h}_0 \sim Gaussian(h|\mu,\sigma) \tag{7}$$

$$h_t = GeneratorRNN(h_{t-1},\hat{v_t}) \quad t = \{1,\dots,T\} \tag{8}$$

$$f_t = W^{\text{Factors}}\cdot h_t \tag{9}$$

$$r_t = \exp(W^{\text{Rates}}\cdot f_t) \tag{10}$$

$$\hat{x_t} \sim \text{Poisson}(x_t|r_t) \tag{11}$$

This LFADS model can be trained using self-supervision learning by minimizing the negative Poisson log-likelihood reconstruction loss and Kullback–Leibler (KL) divergence:

$$\mathscr{L} = \sum_{t=1}^{T} \log(\text{Poisson}(x_t|r_t) + KL(h||z), \tag{12}$$

where z stands for the stand Gaussian noise (other details of the LFADS model can be found in [8]). LFADS can be treated as a robust neural feature extractor, i.e., compressing the raw spike signal into lower-dimensional representations (also known as the latent factors f_t) to model the brain state and neural dynamics. These factors are then used as feature vectors for various downstream tasks tailored to the experiment goals. Ongoing work also uses estimated latent factors to train a simple Multi-Layer Perceptron (MLP) for the downstream regression analysis for the hand movement trajectory prediction.

As with many RNN architectures, LFADS is an iterative time-series sequential model that unrolls the input $X \in \mathbb{R}^{C\times T}$ across T timesteps, producing a predicting firing rate for all C channels at each step. Sequential unrolling allows the model to capture temporal dynamics in the raw neural signal, which significantly increases the inference latency, scaling poorly as we further increase the available signal window. This variable inference latency may cause critical issues in closed-loop systems with limited tolerances for decoding latency, leading to the potential necessity for FPGA acceleration.

C. Dual Operating Systems

The host machine runs Ubuntu 18.04 LTS as its primary operating system (OS) and a virtualized Windows 10 guest OS as a Kernel-based Virtual Machine (KVM). This dual-OS setup is necessary due to hardware and software constraints: the IMEC Neuropixels hardware and its SpikeGLX API are natively supported only on Windows, while interfacing with the FPGA requires the XRT driver, which is only available on Linux. First, the Windows guest OS collects preprocessed data from the IMEC acquisition card with the SpikeGLX API, storing it in a buffer in virtual memory. The buffer is then transferred from the guest to the host domain through VirtIO Queues, which are managed by a host application running on the Ubuntu OS. The host application then formats the data and initiates data transfers with the FPGA using XRT system calls.

D. Datasets for System Evaluation

To evaluate the performance of the co-designed system, we applied the testbed to data collected in vivo at the Orsborn lab of the University of Washington, Washington National

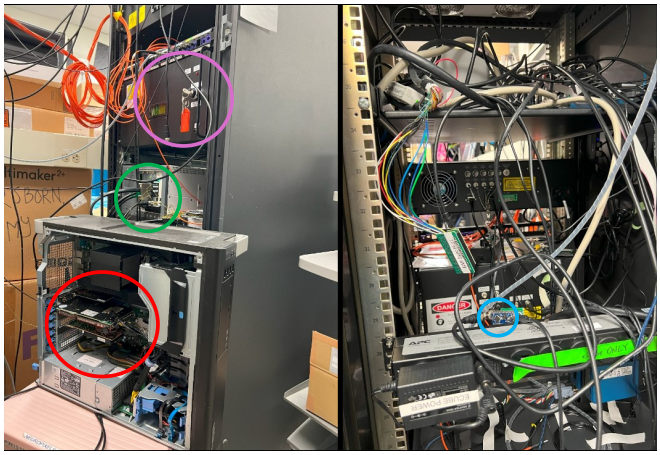

Fig. 2. The built closed-loop BCI system according to the Figure 1. Essential components are circled for better visualization. The Host computer is in red; The IMEC data acquisition module is in green; The Laser generator is in purple; The Arduino laser control board is in blue.

Primate Research Center. Recorded with Neuropixels probes targeting the motor cortex of an NHP, the dataset contains a total of 789 trials - each with $T = 20$ time bins and $C = 105$. The LFADS model was trained and evaluated on the data with $70\%, 15\%, 15\%$ splits for training, validation, and evaluation respectively. Then, the system performance was characterized under a simulated real-time experiment using pre-recorded data files (see *Results* section).

III. RESULTS

We presented the overall system (see Figure 2) built and integrated in our laboratory as described by the methods figure diagram (see Figure 1). The IMEC PXIe acquisition module and the NI PXI 1071 chassis were housed in the server rack to the left of the image and sit above the laser generator and the Arduino laser controller. The host machine was at the figure's right housing a NI 8381 PXIe card and the Xilinx U55C FPGA.

A key performance metric in evaluating the system was the round-trip latency between an input signal from the brain to a consequent stimulus. To characterize the latency in the system, we partitioned the system into multiple data pathways with individual component latencies (see Table 1). The total system latency was calculated as the sum of the components and validated with a separate measurement of the complete system latency. Real-time BCI systems are usually limited by the neural decoding latency. However, by accelerating our signal processing pipeline on the FPGA we reduced our decoding model inference time to a marginal fraction of the system latency with an average inference latency of 0.156ms. We also observed that the system latency was now dominated in data handling and pre-processing by the IMEC acquisition module, which could not be bypassed due to the system design. To this end, our system achieve a promising total latency of 9.351ms.

An additional key criterion in evaluating our co-designed BCI system is the system power consumption - a critical design consideration for developing future portable and long-term usage BCI devices. We compared the average power usage of the FPGA to a CPU and GPU while executing

TABLE I

LATENCY BREAKDOWN IN PROPOSED BCI SYSTEM

Component	Latency (ms)
Headstage & Module Preprocessing	2.512
Module to Host	5.047
Host to FPGA	0.018
Kernel Execution	0.156
FPGA to Host	0.008
Host to Laser Control Board	1.610
Total	9.351

inference on the LFADS model across 10000 batches of data. The power consumption of the FPGA was measured as the total system power consumption reported by XRT driver tools; the CPU power was measured as the CPU package power; and the GPU power was measured as the GPU core input power. Both CPU and GPU power figures were reported by HWiNFO, a tool that monitors onboard power, thermal, and frequency sensors through hardware drivers. To isolate the contribution of the inference algorithm to the power consumption and account for background processes, we derive an active power draw for all three architectures by subtracting their respective idle and runtime power consumptions. We observed that the average power consumption over tested 10000 batches on the FPGA was 27.867 W, utilizing only 33.03% and 31.89% of the power compared to the CPU and GPU (power cost: CPU 84.345 W, GPU: 87.393 W), respectively. We also compared the average inference latency of the LFADS model on the three architectures, finding a substantial 1650x and 118x reduction in inference latency on the FPGA when compared to the CPU and GPU respectively. We also noted that the standard deviation of the inference latency on the FPGA was small with only 0.002ms, showing the robustness of inference speed on FPGA. On the contrary, the CPU and GPU experienced larger deviations, around 56.693ms and 2.995ms respectively. Note that inference on the CPU is not well parallelized by the Tensorflow framework, stressing only 2 of the 16 CPU cores during our tests. In conclusion, our results highlighted the computational and power efficiency of FPGA-accelerated algorithms.

TABLE II

INFERENCE LATENCY & POWER ACROSS ARCHITECTURES

Architecture	Inference Latency (ms)	Active Power Draw (W)
FPGA	0.156 ± 0.002	12.447
CPU†	257.516 ± 56.693	39.490
GPU‡	18.551 ± 2.995	52.606

† AMD 5950x 16-Core CPU
‡ NVIDIA RTX 3090 GPU

Last, we compared the neural decoding performance. Prior investigations highlighted the superior performance in system latency and power consumption, which can partially be explained by our quantized LFADS model with a reduced 8-bit integer representation of weights on the FPGA. However, using the 8-bit integer quantized model, our neural decoding performance still maintained in a reasonable range (ours: 55.21% v.s. full-precision: 61.18%). Together, by testing the performance of our proposed co-design system, we showed that our system enables reasonable neural decoding performance, lower power consumption, and significantly lower total system latency - paving the way for future FPGA-accelerated BCI designs.

IV. DISCUSSIONS AND CONCLUSIONS

In our work, we proposed a co-design of the hardware and software system for the future integrated BCIs. To the best of our knowledge, thisis the first time to have such a comprehensive integration of hardware and software BCI systems. We evaluated the essential system latency, power consumption, and neural data processing results via the FPGA integrating neural decoding model in our proposed BCI system. Our results illustrated the successful neural data communication between the hardware and software systems. Meanwhile, we showed that our BCI system enables ultra-fast processing speed with parsimonious power utilization. Last, we showed that the integration on the hardware FPGA system (under 8-bit integer quantization) also maintains high neural processing decoding performance compared to the full-precision model in software. Our co-design framework paves the way for future implementations of integrated BCI systems.

Our work has several limitations. First, the high-density neural recording technology in our co-design framework is the Neuropixel 1.0 system. Many current BCI implementations use other recording technologies, such as EEG [15], and multielectrode arrays [12]. Exploring the possibility of integrating various neural recording technologies with our BCI system will be an interesting future direction. Second, we proposed an alternative, optical stimulation strategy to close the loop. It should be noticed that using optogenetics as a novel stimulation technique to uncover brain functions or neural mechanism foundations requires sophisticated biological experimental design, which is beyond the scope of this work. Future work will test our optogenetics-based closed-loop BCI systems in regulating nonlinear neural dynamics [16]. Last, we deployed the well-established LFADS as the neural signal processing model on FPGA. Another interesting future direction is to explore the possibility of transformer-based neural predictive models [17] for integration in FPGA.

V. ACKNOWLEDGMENTS

This work was partly supported by the Accelerated AI Algorithms for Data-Driven Discovery (A3D3) Institute under U.S. National Science Foundation Grant No. 2117997.

REFERENCES

[1] J. Robinson *et al.*, "An application-based taxonomy for brain-computer interfaces," *Nat. Biomed. Eng*, pp. 1–3, 2024.

[2] U. Chaudhary, N. Birbaumer, and A. Ramos-Murguialday, "Brain–computer interfaces for communication and rehabilitation," *Nat. Rev. Neurol*, vol. 12, no. 9, pp. 513–525, 2016.

[3] O. G. Sani *et al.*, "Mood variations decoded from multi-site intracranial human brain activity," *Nat. Biotechnol*, vol. 36, no. 10, pp. 954–961, 2018.

[4] M. M. Shanechi, "Brain-machine interfaces from motor to mood," *Nat. Neurosci.*, vol. 22, no. 10, pp. 1554–1564, 2019.

[5] A. L. Orsborn *et al.*, "Closed-loop decoder adaptation shapes neural plasticity for skillful neuroprosthetic control," *Neuron*, vol. 82, no. 6, pp. 1380–1393, 2014.

[6] K. W. Scangos *et al.*, "Closed-loop neuromodulation in an individual with treatment-resistant depression," *Nat. Med.*, vol. 27, no. 10, pp. 1696–1700, 2021.

[7] H. Fang *et al.*, "Emotion recognition from eeg network connectivity using low-dimensional discriminant analysis on Riemannian manifolds," *Authorea Preprints*, 2024.

[8] C. Pandarinath *et al.*, "Inferring single-trial neural population dynamics using sequential auto-encoders," *Nat. Methods*, vol. 15, no. 10, pp. 805–815, 2018.

[9] C. Chen *et al.*, "Model-agnostic meta-learning for EEG-based inter-subject emotion recognition," *J. Neural Eng.*, 2024.

[10] H. Fang and Y. Yang, "Designing and validating a robust adaptive neuromodulation algorithm for closed-loop control of brain states," *J. Neural Eng.*, vol. 19, no. 3, p. 036018, 2022.

[11] H. Fang *et al.*, "Robust adaptive deep brain stimulation control of in-silico non-stationary parkinsonian neural oscillatory dynamics," *J. Neural Eng.*, vol. 21, no. 3, p. 036043, 2024.

[12] M. S. Willsey *et al.*, "A high-performance brain–computer interface for finger decoding and quadcopter game control in an individual with paralysis," *Nat. Med.*, pp. 1–9, 2025.

[13] K. W. Scangos *et al.*, "State-dependent responses to intracranial brain stimulation in a patient with depression," *Nat. Med.*, vol. 27, no. 2, pp. 229–231, 2021.

[14] J. Duarte *et al.*, "Fast inference of deep neural networks in fpgas for particle physics," *JINST*, vol. 3, no. 07, p. P07027, 2018.

[15] F. Lotte *et al.*, "A review of classification algorithms for eeg-based brain–computer interfaces: a 10 year update," *J. Neural Eng.*, vol. 15, no. 3, p. 031005, 2018.

[16] B. Zaaimi *et al.*, "Closed-loop optogenetic control of the dynamics of neural activity in non-human primates," *Nat. Biomed. Eng*, vol. 7, no. 4, pp. 559–575, 2023.

[17] T. Le and E. Shlizerman, "Stndt: Modeling neural population activity with spatiotemporal transformers," *Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 17 926–17 939, 2022.