nature medicine

LETTERS

https://doi.org/10.1038/s41591-018-0171-y

Meeting brain–computer interface user performance expectations using a deep neural network decoding framework

Michael A. Schwemmer[1]*, Nicholas D. Skomrock[1], Per B. Sederberg[2], Jordyn E. Ting[3], Gaurav Sharma[3], Marcia A. Bockbrader[4,5,6] and David A. Friedenberg[1,6]

Brain–computer interface (BCI) neurotechnology has the potential to reduce disability associated with paralysis by translating neural activity into control of assistive devices[1–9]. Surveys of potential end-users have identified key BCI system features[10–14], including high accuracy, minimal daily setup, rapid response times, and multifunctionality. These performance characteristics are primarily influenced by the BCI's neural decoding algorithm[1,15], which is trained to associate neural activation patterns with intended user actions. Here, we introduce a new deep neural network[16] decoding framework for BCI systems enabling discrete movements that addresses these four key performance characteristics. Using intracortical data from a participant with tetraplegia, we provide offline results demonstrating that our decoder is highly accurate, sustains this performance beyond a year without explicit daily retraining by combining it with an unsupervised updating procedure[3,17–20], responds faster than competing methods[8], and can increase functionality with minimal retraining by using a technique known as transfer learning[21]. We then show that our participant can use the decoder in real-time to reanimate his paralyzed forearm with functional electrical stimulation (FES), enabling accurate manipulation of three objects from the grasp and release test (GRT)[22]. These results demonstrate that deep neural network decoders can advance the clinical translation of BCI technology.

The study participant was a 27-year-old male with C5 AIS A tetraplegia due to spinal cord injury. He was implanted with a 96-channel microelectrode array in the hand and arm area of his left primary motor cortex[8,23,24]. We trained and evaluated BCI decoders using 80 sessions of intracortical data collected from the participant over 865 d. During each session, the participant performed two 104-s blocks of the four-movement task (Methods), in which he was cued to imagine a series of four distinct hand movements (index extension, index flexion, wrist extension, wrist flexion) in a random order (Fig. 1a,b).

We calibrated the initial neural network (NN) model using 40 sessions (80 blocks) from the training period (Fig. 1c). As the model was not updated at all over the subsequent test period, we call it the fixed NN (fNN). Two additional NN models were created from the fNN using updating procedures that used the first block of each of the 40 sessions in the testing period in different ways (Fig. 1c): supervised updating (sNN) or unsupervised updating (uNN) (Fig. 1d and Methods). In this context, supervised refers to the algorithm using explicit training labels (i.e., known timing and type of intended action) as opposed to unsupervised, in which the timing and type of intended action were unknown, as occurs with general BCI use. For comparison, the first block of each of the 40 sessions in the testing period was also used to calibrate benchmark BCI decoders that were retrained daily: a support vector machine (SVM) decoder (Fig. 1d)[8,23,25,26], a linear discriminant analysis (LDA) decoder[17], and a naive Bayes decoder[18]. The SVM performed better than the LDA or naive Bayes decoder (Supplementary Fig. 1) and was thus used for further comparisons with NN performance. Neural features used by all models were the mean wavelet power (MWP) values calculated from raw voltage for each of the 96 channels over 100-ms bins[8,23,24] (Fig. 1e, Supplementary Fig. 2, and Methods). Performance for each of the NN and comparison models was initially evaluated using accuracy (percentage of correctly predicted time-bins) on the second block of data from each session during the testing period (Methods). Figure 1e shows data processing steps and NN model architecture.

To quantify improved BCI accuracy with the NN, we compared the performance of the supervised, daily-updated sNN against a daily-retrained SVM. Figure 2a,b shows that the sNN was more accurate than the daily-retrained SVM for all sessions, with a mean difference of $6.35 \pm 2.47\%$ (mean $\pm$ s.d.; $P = 3.69 \times 20^{-8}$, $V = 820$, $n = 40$ paired two-sided Wilcoxon signed rank test; n is the sample size and V is the test statistic for the paired Wilcoxon test). In addition, for 37 out of 40 sessions, the sNN accuracy was >90%, indicating consistently high performance in accordance with user expectations[13]. In contrast, the SVM accuracy was >90% for only 12 sessions.

To demonstrate that a BCI with a neural network decoder (NN-BCI) could sustain high accuracy for over a year without the need of supervised updating (thus reducing daily setup time), we evaluated performance of the fNN. Figure 2c shows that the fNN was more accurate than the daily-retrained SVM for 36 out of 40 test sessions, with a mean difference of $4.56. \pm 3.06\%$ ($P = 1.90 \times 10^{-7}$, $V = 798$, $n = 40$; Fig. 2c, inset). In addition, the fNN accuracy was > 90% for 32 sessions. Not only was the fNN able to sustain high accuracy-decoding performance for over a year (381 d) without being recalibrated, it significantly outperformed all fixed versions of the benchmark decoders we tested (Supplementary Fig. 1). However, the fNN accuracy was lower than that of the sNN (Fig. 2d), which received supervised updates throughout the testing period. In fact,

[1]Advanced Analytics, Battelle Memorial Institute, Columbus, OH, USA. [2]Department of Psychology, University of Virginia, Charlottesville, VA, USA. [3]Medical Devices and Neuromodulation, Battelle Memorial Institute, Columbus, OH, USA. [4]Neurological Institute, Ohio State University, Columbus, OH, USA. [5]Department of Physical Medicine and Rehabilitation, Ohio State University, Columbus, OH, USA. [6]These authors contributed equally: Marcia A. Bockbrader, David A. Friedenberg. *e-mail: schwemmer@battelle.org

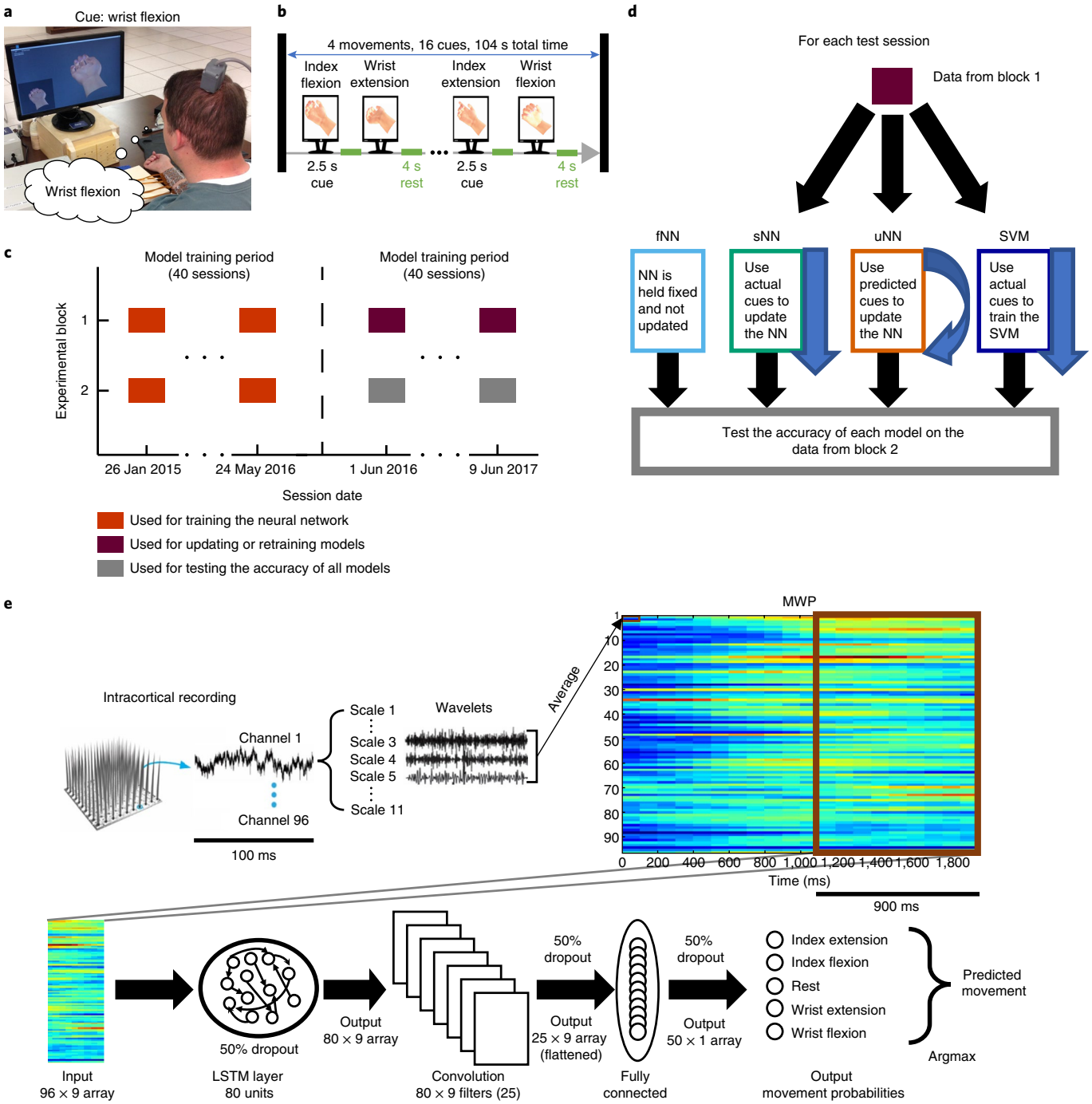

Fig. 1 | Experimental set-up, data processing steps, and NN architecture. a, The participant performing the four-movement task. The movement is displayed on the computer screen by a virtual hand and he imagines performing the movement, without receiving visual or movement feedback. **b**, Structure of one block of the four-movement task. Each of the four movements (index flexion and extension and wrist flexion and extension) are repeated four times for a duration of 2.5 s in a random order, with 4 s of rest following each cue. **c**, Layout of how the historical four-movement task data is used for the training and testing of the decoding methods. Each session consists of two blocks of the task, and the first 40 sessions (80 blocks, 2.31 h of data) are used to train the NN. The subsequent 40 sessions are the testing period, during which the first block can be used for updating and retraining a decoding model, whereas the second block is always held out to test model accuracy. The training period spanned a total of 484 d, the testing period spanned 373 d, and the time from the last training day to the last testing day was 381 d. **d**, How each of the four decoding methods use the data for a given test session. The fNN is trained on the 80 blocks from the training period and not updated. The sNN and uNN both start as the fNN but are updated using the data from the first block of a test session in either a supervised or an unsupervised manner (Methods). The SVM decoder is retrained every test session using only the data from the first block of that session. **e**, Schematic of the data processing and NN model architecture. Raw voltage recordings from each of the 96 channels were first processed in 100-ms segments. Wavelet decomposition was applied to the data, and the coefficients from scales 3, 4, and 5 were averaged across time and then across scales in order to produce a single value, MWP, for each channel for each 100-ms time bin. The NN then takes as input a sliding 900-ms window of the MWP for each of the 96 channels (a 96×9 array) in order to predict the intended movement for the current time point. The NN consists of a long short-term memory (LSTM) layer, a convolutional layer with twenty-five 80×9 filters, a dense layer with 50 units, and an output layer that produces a vector of five probability values (four movements and rest). The movement with the highest probability value (i.e., the argmax of the predicted movement probabilities) is the predicted movement for that 100-ms time bin.

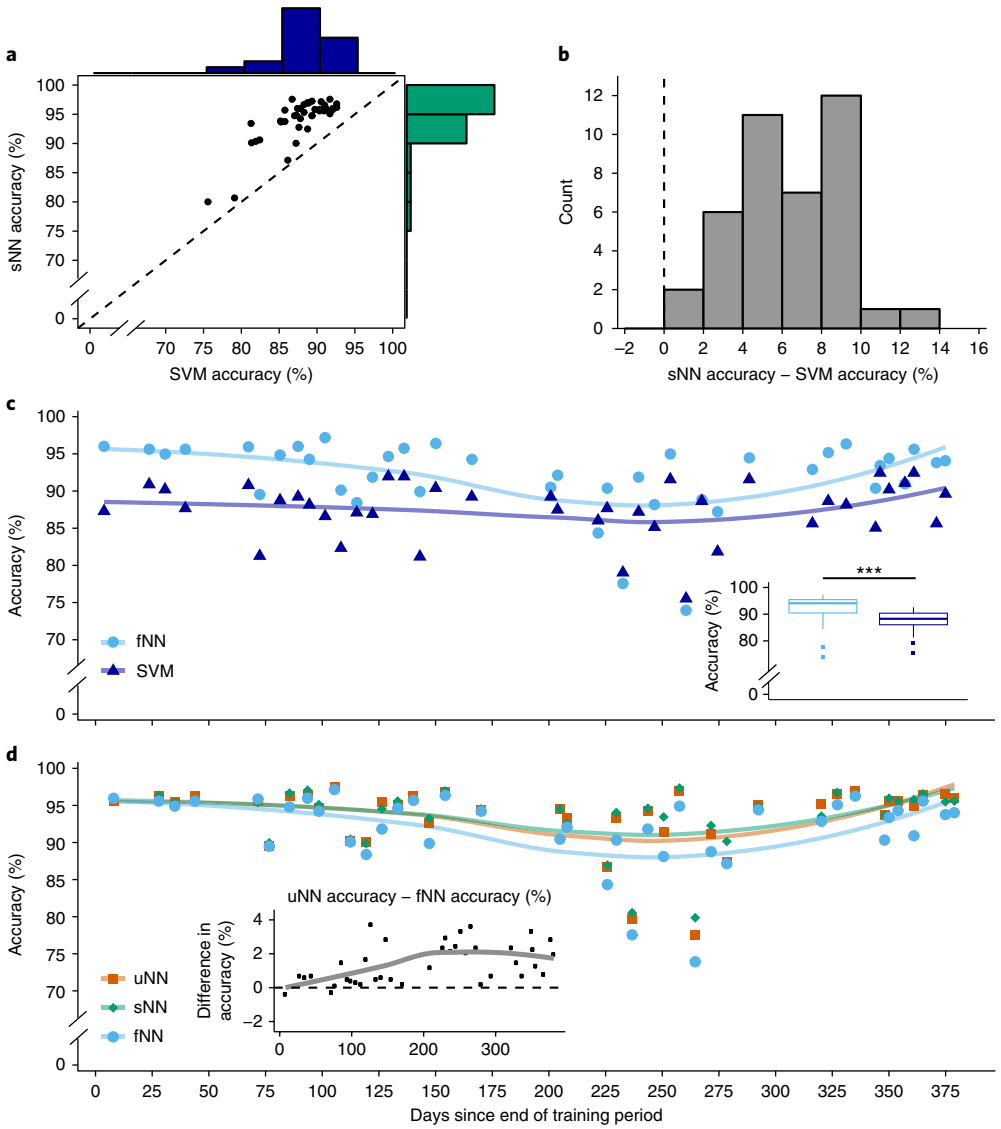

Fig. 2 | Year-long high-fidelity decoding of movement intentions with NNs. a, sNN accuracy for each of the 40 test sessions plotted as a function of the corresponding SVM accuracy. Marginal histograms of the model accuracies are shown in the upper and right-most portions of the plot. **b**, Histogram of the paired differences in accuracy (sNN – SVM) for each of the test sessions. The vertical dashed line denotes zero difference in accuracy. **c**, fNN (cyan circles) and SVM (dark blue triangles) accuracies are plotted as a function of the number of days since the end of the NN training period. The lines denote a LOESS smoothing curve to visualize the data trends. The inset shows a boxplot of the accuracies of the two models, where the thick horizontal line denotes the median, the lower and upper hinges correspond to the first and third quartiles, the whiskers extend from the hinge to the most extreme value no further than 1.5× interquartile range from the hinge, and the dots beyond the whiskers are outliers. The stars denote that the difference in accuracy is significant ($P = 1.9 \times 10^{-7}$, paired Wilcoxon signed rank test; $n = 40$, $V = 798$). **d**, The accuracies of the NNs with either supervised (sNN, green diamonds) or unsupervised (uNN, orange squares) updating are plotted along with the fNN accuracy (cyan circles). The inset plots the paired differences in accuracy (uNN – fNN) for each session during the test period. All models had their worst performance on post-training period days 238 and 266; these dates corresponded to times when the participant had undiagnosed infections

the performance of all fixed decoders declined over time relative to their daily-retrained versions (Supplementary Fig. 3). This difference was likely due to the fixed decoders not being able to adapt to changes in the underlying neural data patterns across time[17,19,27]. This observation motivated the use of our unsupervised updating method, which allows the decoder to adapt without requiring the user to dedicate time for collection of additional supervised calibration data.

To show that a NN-BCI could maximize and maintain accuracy by leveraging data generated as the participant used the system, we evaluated uNN performance. In contrast to the sNN, in which explicit training with knowledge of the type and timing of the movement cues was required for updating, the uNN inferred these aspects of movements from the neural data (Methods). Thus, the unsupervised updating can be performed with data collected during general use, whereas the supervised updating requires collection of calibration data that contains the type and timing of intended movements, requiring undesirable setup time. Figure 2d shows that unsupervised updating boosted uNN accuracy above fNN accuracy, with a mean difference of $1.56 \pm 1.36\%$ ($P = 9.76 \times 10^{-8}$, $V = 807$, $n = 40$) and a mean improvement in accuracy of $6.12 \pm 2.68\%$ over the daily-retrained SVM (confusion matrices for the sNN, fNN, uNN, and SVM are shown in Supplementary Fig. 4). The Fig. 2d inset shows that the performance gap between the fNN and the uNN grew as

a

Model	Failure rate (missed cues)				
	Index extension	Index flexion	Wrist extension	Wrist flexion	All
SVM	25.66% (39)	31.61% (49)	14.97% (22)	15.75% (23)	22.17% (133)
uNN	6.58% (10)	6.45% (10)	6.12% (9)	3.42% (5)	5.67% (34)

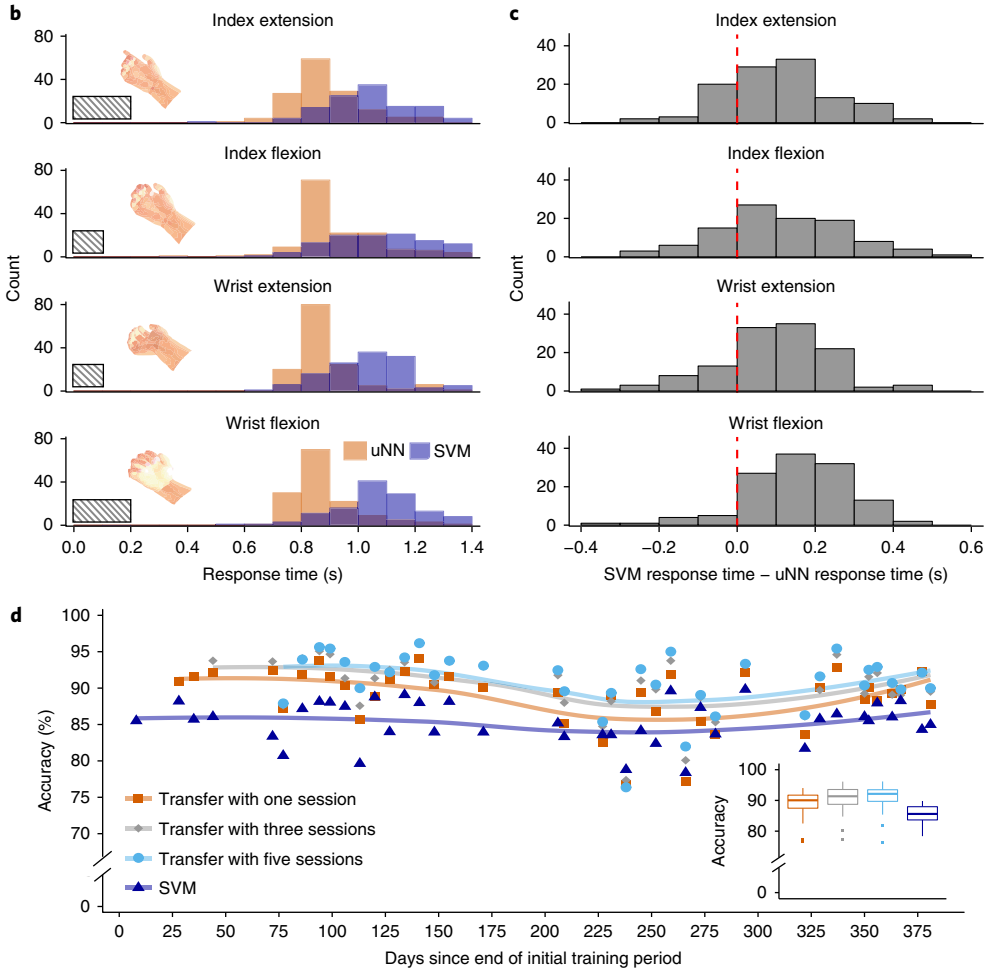

Fig. 3 | Translating gains in NN accuracy to system usability and increasing the number of available functions with transfer learning. a, The failure rate (percentage of cues for which the model failed to predict the correct cue and sustain it for at least 1 s) is shown for the individual movements for the SVM and uNN decoders. Note that there are 600 total cued movements, as the first cued movement is ignored in each session because the uNN starts predicting during the middle of cue because of the time-lagged input. Thus, the number of cues for each movement is: 152 for index extension, 155 for index flexion, 147 for wrist extension, and 146 for wrist flexion. **b**, Histogram of the time it takes for the model responses, measured as the time from cue onset to the decoder predicting the correct cue and sustaining it for at least 1 s (SVM: dark blue; uNN: orange). The hashed bars show the time it takes from cue onset for the virtual hand to complete the cued movement animation. Response times were constrained to be multiples of 0.1 s since the decoders were updated in 0.1 s segments. **c**, Histogram of the paired differences in response times (SVM – uNN) for those movements for which both models responded correctly (index extension: $n = 112$; index flexion: $n = 104$; wrist extension: $n = 120$; wrist flexion: $n = 123$). The red vertical line denotes zero difference in response time. **d**, Accuracy following an increased number of available movements using transfer learning and additional historical data. Data from the four- and two-movement tasks are combined to create a new six-movement task (Supplementary Fig. 6 and Methods). The accuracy of the utNN models trained using either one, three, or five sessions' worth of data (orange squares $n = 39$, gray diamonds $n = 37$, and cyan circles $n = 35$) and SVM (dark blue triangles $n = 40$) are plotted as a function of the number of days since the end of the initial NN training period. The number of sessions translates to 6.7 min (one), 20.2 min (three), or 33.7 min (five) of total data collection time. The inset shows a boxplot of the accuracies of the four models, where the thick horizontal line denotes the median, the lower and upper hinges correspond to the first and third quartiles, the whiskers extend from the hinge to the most extreme value no further than 1.5 × interquartile range from the hinge, and the dots beyond the whiskers are outliers

the testing period progressed. That is, the performance benefits of unsupervised updating increased over time.

To better understand how gains in NN accuracy relate to system usability, we determined how accuracy impacted NN-BCI failure rate and speed. Failure rate was defined as the percentage of cues for which the algorithm failed to predict and sustain the correct movement for at least 1 s. Figure 3a shows that the higher accuracy of the uNN translated to a lower failure rate (5.67%) than that of the daily-retrained SVM (22.17%) across all movements (success rate = 100% – the failure rate; calculated for the uNN and SVM for each session during the testing period and shown in Supplementary Fig. 5). BCI speed was determined by the response time, the lag between

cue onset and initiation of the correct decoder activation that is sustained for at least 1 s. Figure 3b shows that the median uNN response time (0.9 s) was faster than the median SVM response time (1.1 s) across all movements. By comparing paired differences between uNN and SVM response time for correctly decoded trials, we found (Fig. 3c) that the uNN responded significantly faster than the SVM for all movements, with a median improvement of 0.2 s (all $P < 1 \times 10^{-10}$; index extension: $n = 112$, $V = 4{,}138.5$; index flexion: $n = 104$, $V = 3{,}683.5$; wrist extension: $n = 120$, $V = 5{,}313$; wrist flexion: $n = 123$, $V = 6{,}702.5$). Note that the 0.9-s median response time for the uNN is not related to the size of the time-lagged input to the model (Supplementary Fig. 5). Therefore, the 6.12% increase in accuracy of the uNN relative to the SVM translated into a potentially clinically significant 200-ms improvement in response time and a 16.5% improvement in failure rate.

To show that functionality of a NN-BCI could be extended with minimal impact on setup and training time, we used transfer learning[21] to increase the number of movements decoded by the fNN from four to six. In transfer learning, the learned parameters of a NN model initially trained on one task are repurposed (transferred) to a new related task. By leveraging the parameters learned for the initial task, the transfer model can be calibrated using far less data than the initial model. The implication for a NN-based BCI system is the potential to increase the number of movements with minimal calibration required for the additional movements. Using this approach, new NN decoders were built for six movements, adding hand open and hand close to the four movements in the original fNN. The new models used only one, three, or five blocks of supervised data for the new movements from the training period of the transfer model (Supplementary Fig. 6 and Methods). Unsupervised updating was also applied to these models during the transfer testing period, and we refer to them as utNNs. Adding one session (6.7 min of calibration data) to learn new movements using the transfer model yielded an average accuracy of 88.89 ± 4.18% ($n = 39$) across the transfer testing period. Using three or five training blocks resulted in average accuracy of 90.37 ± 4.32% ($n = 37$) or 90.97 ± 4.27% ($n = 35$), respectively. In comparison, SVM accuracy was 85.44 ± 4.14% ($n = 40$). Across the 1-year testing period (Fig. 3d), accuracy of the utNN with only a single session of additional training data was significantly better than the SVM, with an average difference of 3.34 ± 2.70% ($P = 1.01 \times 10^{-7}$, $V = 601$, $n = 39$) demonstrating that transfer learning can be used to efficiently add additional functionality to existing NN decoders.

To demonstrate that offline uNN training could generalize to real-time control of functional hand grasps by a BCI user with paralysis, we performed an experiment in which the participant used a transferred version of the uNN decoder to control FES of his paralyzed forearm (Methods). That is, we used transfer learning on the uNN model previously trained on the offline four-movement task to create a new decoder that predicted hand open and the three grips (peg, fork, and can from the standardized GRT[22]) shown in Fig. 4a,b. Supervised updating (~7.27 min over three blocks) was used to transfer the uNN to these new movements. The GRT was then administered by a board-certified physiatrist for these three objects (Fig. 4c–e; Supplementary Videos 1 and 2 show the participant manipulating GRT objects using this decoder to control FES of his paralyzed forearm, and Supplementary Video 3 shows a sample GRT run). Without the system, the participant was unable to manipulate the can or fork (median of zero transfers in 30 s for both) and transferred the peg using an adaptive grip (median of five transfers in 30 s). With the system, the participant consistently performed four can transfers, five peg transfers, and six fork manipulations in each of three trials. He had no drops or failures in 45 attempts. Average times to transfer the can (which required sequential activation of hand open and grip) and depress the fork were: 6.39 ± 0.84 s ($n = 12$) and 3.62 ± 0.85 s ($n = 18$), respectively. Peg transfer times were not significantly different from his adaptive-grip baseline ($P = 0.29$, $W = 93$, $n_{System\ off} = 16$, $n_{System\ on} = 15$; unpaired Wilcoxon test where W is the test statistic).

Optimization of BCI-enabled neuroprosthetics to reflect end-user design priorities and performance expectations can expedite clinical translation of investigational systems, facilitate testing against clinically relevant design standards, and maximize the likelihood of acceptance and adoption by end-users. We focused on BCI system priorities identified by potential end-users with paralysis, specifically accuracy, set-up time and sustainable performance, response time, and the number of available functions[12–14], all of which pose considerable barriers to daily use of BCIs as assistive devices. As suggested by Ajiboye et al.[9], Bouton et al.[8], and others, we have shown that improving the neural decoder component of the BCI system is a critical step in meeting performance expectations of BCI users for home use.

The technical advances of our results include: (i) using deep NNs to create robust neural decoders that sustain high fidelity BCI control for more than a year without retraining; (ii) introducing a new updating procedure that can improve performance using data obtained through regular system use; (iii) extension of functionality through transfer learning using minimal additional data; and (iv) introducing a decoding framework that simultaneously address these four competing aspects of BCI performance (accuracy, speed, longevity, and multifunctionality). In addition, we provide a clinical demonstration that a decoder calibrated using historical data of imagined hand movements with no feedback can be successfully used in real-time to control FES-evoked grasp function for object manipulation.

The high degree of variability in neural activity both within and across recording days[3,27–29] is a considerable barrier to creating accurate and robust neural decoders for BCI systems. Failure to account for this variability causes a steady decline in decoder performance over time[19,27,30]. Mitigating this decline by recalibrating the decoder before every use[8,19,27,30] leads to an undesirable increase in setup time. Several approaches have been proposed to avoid daily recalibration while also making neural decoders more robust to day-to-day variability[3,17,18,31–34]. One set of approaches is aimed at using data collected during practical use in order to continuously adapt the parameters of the decoder to changing recording conditions[3,17,18,32]. Studies using these methods have shown sustained performance for at most a year[3,7,17]. However, these studies used linear decoding methods (for example, naive Bayes or LDA classifiers), which can be less accurate than more complex nonlinear methods[25,35]. Alternatively, Sussillo et al.[33] demonstrated that recurrent NNs calibrated on a large amount of historical primate data (up to 50 sessions consisting of ~500 trials) outperformed a simpler daily recalibrated decoder for computer cursor control and sustained performance for up to 59 d without recalibration. However, their decoder relied solely on historical data to learn a static mapping between neural activity and intended action, similar to our fNN. Like our fNN, their decoder appeared to decline in performance over time, albeit at a slow pace.

Our innovative approach, which achieved 93.78 ± 4.39% accuracy and missed only 5.67% of cues across 381 d, combines aspects of these NN and adaptive updating methodologies. Importantly, our updating procedure requires no intervention from the user, and makes no assumptions about the task other than it can be formulated as a discrete classification problem. In contrast, a daily-recalibrated SVM decoder built from the same data[8,23,24,26] had an average accuracy of 87.66 ± 3.85% and missed 22.17% of the total cues. Thus, the uNN simultaneously approached the 90% user benchmark for accuracy[13], eliminated daily decoder setup[12], and enabled naturalistic control of objects during a standardized, functional transfer task (transfer time range: 3.62–6.39 s). Using a different BCI system with an implanted FES device, the participant from Ajiboye et al.[9] had a 61% success rate with 3 movements and a mean movement time

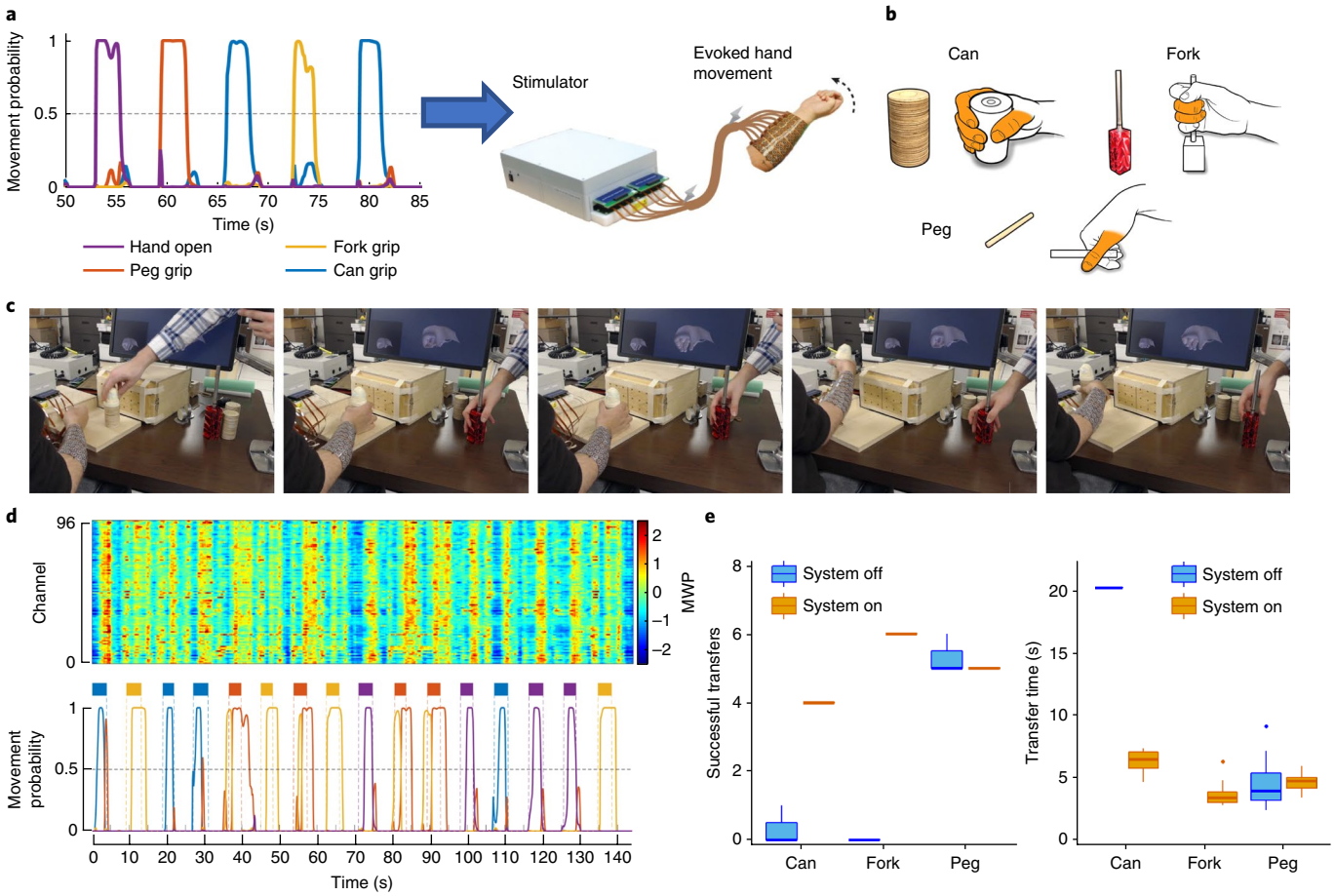

Fig. 4 | Real-time control of functional electrical stimulation. a, The output of the NN decoder (i.e., the movement with highest predicted probability) was used to control FES of the participant's paralyzed forearm. The participant wore a flexible cuff consisting of 130 electrodes which can be seen in **c**. **b**, The can, fork, and peg objects from the GRT along with the associated grasps enabled by the FES system. The objects have dimensions 9.1 × 5.4 (diameter) cm for can, 7.6 × 0.6 (diameter) cm for peg, and 14.5 × 1.2 (diameter) cm for fork. **c**, Example of the cued movement for can. Prior to the cue, the object is placed in front of the participant. He is then cued to grip the object and move it to an elevated platform. The object is then removed before the next object being place in front of him. **d**, Example cued block in which the participant was in control of the FES system. The figure shows the output of the NN decoder (bottom) along with the MWP features for each of the 96 channels. The cued movement times are shown as solid color blocks (top). MWP values were capped at 2.5 for better visualization. A video recording of this block is provided in Supplementary Video 1. **e**, The participant's performance in terms of the number of successful transfers (left) and the transfer times for each object (transfers: $n = 3$ for all objects for both system on and off; transfer times: $n = 16$ for peg, $n = 0$ for fork, and $n = 1$ for can with the system off and $n = 15$ for peg, $n = 18$ for fork, and $n = 12$ for can with the system on) over three runs of the GRT. During each run, he was required to grasp, transfer the object to an elevated platform, and release it as many times as possible in a 30 second test window. System off is the participant's baseline performance without the use of the BCI-FES system. In the boxplots, the thick horizontal line denotes the median, the lower and upper hinges correspond to the first and third quartiles, the whiskers extend from the hinge to the most extreme value no further than 1.5 × interquartile range from the hinge, and the dots beyond the whiskers are outliers

of 8.6 s. However, the multiple methodological differences between our studies (for example, different levels of participant spinal cord injury, different FES solutions, different experimental conditions, a different number of recording electrodes, different decoder outputs, and different performance metrics; Methods) makes direct comparison between our results difficult.

Although the median uNN response time (0.9 s) was faster than that of the SVM (1.1 s) for the same training set, it is not clear what response times are acceptable to users[12]. BCI experiments with able-bodied users suggest a goal of under 750 ms, as sense of agency—the feeling of being in control of the BCI system and an important factor in user acceptance—decreases as response time increases beyond that point[36]. Multifunctionality is also a priority of potential BCI users, though the optimal number of functions may be specific to the BCI application. In particular, BCI systems controlling robotic arms may need to independently control 10 degrees of freedom[37], whereas others require discrimination of two states, like 'Yes–No' communication devices for patients in a minimally conscious state[38] or grip neuroprosthetics with only hand-open or hand-closed states[39]. We have demonstrated that decoders based on NNs may be superior to other implementations because new functions can be easily added after the initial decoder calibration using transfer learning[21]. Crucially, we show that this secondary update to add more movements requires a minimal amount of additional data.

Finally, we demonstrated that insights gained from offline data and analyses can carry over to a realistic online BCI scenario with minimal additional data collection. Specifically, uNN parameters trained on imagined movements (without visual or movement feedback) were transferrable to real-time control of evoked movements with visual and movement feedback, to manipulate functional objects. It is important to note that the same transfer strategy can generalize beyond FES to control of a robotic arm, computer cursors, and communication or other assistive devices by adjusting the architecture and optimization of the NN (see ref. [33] for an example).

Limitations of this report include: single-participant intracortical data, a training period of relatively long duration (40 sessions across 484 d), the fact that data used for unsupervised updating were collected under controlled lab conditions rather than during spontaneous use, and reliance on offline training and analysis for the base fNN. Supplementary Fig. 7 and Supplementary Tables 1 and 2 show that substantially fewer training sessions still yield viable decoders, and Supplementary Fig. 8 indicates that training the uNN on an equivalent amount of data collected from fewer sessions (shorter timespan) is also possible. However, more work is needed to quantify the minimum number and duration of training sessions needed for stable NN-BCIs. Similarly, because of the limited number of electrodes and the small intracortical volume (16 mm^3) being sampled by the implanted array, there is a practical limit to the number of new functions that can be added through transfer learning, but more work is needed to characterize the limits of this method. To facilitate translation for clinical usage, future work should explore the generalizability of results to other subjects, the practicality and success of performing unsupervised updating using data from unstructured tasks, and the relationship between movements added through transfer learning and accuracy as well as further demonstration of robust and accurate online decoding in the clinical setting.

Online content

Any methods, additional references, Nature Research reporting summaries, source data, statements of data availability and associated accession codes are available at https://doi.org/10.1038/s41591-018-0171-y

Received: 12 December 2017; Accepted: 31 July 2018;
Published online: 24 September 2018

References

1. Lebedev, M. A. & Nicolelis, M. A. L. Brain–machine interfaces: from basic science to neuroprostheses and neurorehabilitation. *Physiol. Rev.* **97**, 767–837 (2017).
2. Chaudhary, U., Birbaumer, N. & Ramos-Murguialday, A. Brain–computer interfaces for communication and rehabilitation. *Nat. Rev. Neurol.* **12**, 513–525 (2016).
3. Jarosiewicz, B. et al. Virtual typing by people with tetraplegia using a self-calibrating intracortical brain–computer interface. *Sci. Transl. Med.* **7**, 313ra179 (2015).
4. Hochberg, L. R. et al. Reach and grasp by people with tetraplegia using a neurally controlled robotic arm. *Nature* **485**, 372–375 (2012).
5. Simeral, J. D., Kim, S.-P., Black, M. J., Donoghue, J. P. & Hochberg, L. R. Neural control of cursor trajectory and click by a human with tetraplegia 1000 days after implant of an intracortical microelectrode array. *J. Neural. Eng.* **8**, 025027 (2011).
6. Collinger, J. L. et al. High-performance neuroprosthetic control by an individual with tetraplegia. *Lancet* **381**, 557–564 (2013).
7. Gilja, V. et al. Clinical translation of a high-performance neural prosthesis. *Nat. Med.* **21**, 1142–1145 (2015).
8. Bouton, C. E. et al. Restoring cortical control of functional movement in a human with quadriplegia. *Nature* **533**, 247–250 (2016).
9. Ajiboye, A. B. et al. Restoration of reaching and grasping movements through brain-controlled muscle stimulation in a person with tetraplegia: a proof-of-concept demonstration. *Lancet* **389**, 1821–1830 (2017).
10. Snoek, G. J., IJzerman, M. J., Hermens, H. J., Maxwell, D. & Biering-Sorensen, F. Survey of the needs of patients with spinal cord injury: impact and priority for improvement in hand function in tetraplegics. *Spinal Cord* **42**, 526–532 (2004).
11. Anderson, K. D. Targeting recovery: priorities of the spinal cord-injured population. *J. Neurotrauma* **21**, 1371–1383 (2004).
12. Collinger, J. L. et al. Functional priorities, assistive technology, and brain-computer interfaces after spinal cord injury. *J. Rehabil. Res. Dev.* **50**, 145–160 (2013).
13. Huggins, J. E., Moinuddin, A. A., Chiodo, A. E. & Wren, P. A. What would brain-computer interface users want: opinions and priorities of potential users with spinal cord injury. *Arch. Phys. Med. Rehabil.* **96**, S38–S45.e5 (2015).
14. Huggins, J. E., Wren, P. A. & Gruis, K. L. What would brain-computer interface users want? Opinions and priorities of potential users with amyotrophic lateral sclerosis. *Amyotroph. Lateral. Scler.* **12**, 318–324 (2011).
15. Kao, J. C., Stavisky, S. D., Sussillo, D., Nuyujukian, P. & Shenoy, K. V. Information systems opportunities in brain-machine interface decoders. *Proc. IEEE Ins. Electr. Electron. Eng.* **102**, 666–682 (2014).
16. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521**, 436–444 (2015).
17. Jarosiewicz, B. et al. Retrospectively supervised click decoder calibration for self-calibrating point-and-click brain-computer interfaces. *J. Physiol. Paris* **110**, 382–391 (2016).
18. Bishop, W. et al. Self-recalibrating classifiers for intracortical brain–computer interfaces. *J. Neural. Eng.* **11**, 026001 (2014).
19. Bacher, D. et al. Neural point-and-click communication by a person with incomplete locked-in syndrome. *Neurorehabil. Neural. Repair.* **29**, 462–471 (2015).
20. Rosenberg, C., Hebert, M. & Schneiderman, H. Semi-supervised self-training of object detection models. in *Seventh IEEE Workshop Appl. Comput. Vis.* (IEEE, Piscataway, NJ, USA, 2005).
21. Yosinski, J., Clune, J., Bengio, Y. & Lipson, H. How transferable are features in deep neural networks? in *Advances in Neural Information Processing* Systems *27* (eds. Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D. & Weinberger, K. Q.) 3320–3328 (Curran Associates, Inc., Red Hook, NY, USA, 2014).
22. Wuolle, K. S., Van Doren, C. L., Thrope, G. B., Keith, M. W. & Peckham, P. H. Development of a quantitative hand grasp and release test for patients with tetraplegia using a hand neuroprosthesis. *J. Hand Surg. Am.* **19**, 209–218 (1994).
23. Sharma, G. et al. Using an artificial neural bypass to restore cortical control of rhythmic movements in a human with quadriplegia. *Sci. Rep.* **6**, 33807 (2016).
24. Friedenberg, D. A. et al. Neuroprosthetic-enabled control of graded arm muscle contraction in a paralyzed human. *Sci. Rep.* **7**, 8386 (2017).
25. Fernández-Delgado, M., Cernadas, E., Barro, S. & Amorim, D. Do we need hundreds of classifiers to solve real world classification problems? *J. Mach. Learn. Res.* **15**, 3133–3181 (2014).
26. Colachis, S. C. et al. Dexterous control of seven functional hand movements using cortically-controlled transcutaneous muscle stimulation in a person with tetraplegia. *Front. Neurosci.* **12**, 208 (2018).
27. Perge, J. A. et al. Intra-day signal instabilities affect decoding performance in an intracortical neural interface system. *J. Neural. Eng.* **10**, 036004 (2013).
28. Aflalo, T. et al. Neurophysiology. Decoding motor imagery from the posterior parietal cortex of a tetraplegic human. *Science* **348**, 906–910 (2015).
29. Pandarinath, C. et al. High performance communication by people with paralysis using an intracortical brain-computer interface. *eLife* **6**, e18554 (2017).
30. Jarosiewicz, B. et al. Advantages of closed-loop calibration in intracortical brain–computer interfaces for people with tetraplegia. *J. Neural. Eng.* **10**, 046012 (2013).
31. Nuyujukian, P. et al. Performance sustaining intracortical neural prostheses. *J. Neural. Eng.* **11**, 066003 (2014).
32. Li, Z., O'Doherty, J. E., Lebedev, M. A. & Nicolelis, M. A. L. Adaptive decoding for brain–machine interfaces through Bayesian parameter updates. *Neural Comput.* **23**, 3162–3204 (2011).
33. Sussillo, D., Stavisky, S. D., Kao, J. C., Ryu, S. I. & Shenoy, K. V. Making brain–machine interfaces robust to future neural variability. *Nat. Commun.* **7**, 13749 (2016).
34. Flint, R. D., Scheid, M. R., Wright, Z. A., Solla, S. A. & Slutzky, M. W. Long-term stability of motor cortical activity: implications for brain machine interfaces and optimal feedback control. *J. Neurosci.* **36**, 3623–3632 (2016).
35. Glaser, J. I. et al. Machine learning for neural decoding. Preprint at https://arxiv.org/abs/1708.00909 (2017).
36. Evans, N., Gale, S., Schurger, A. & Blanke, O. Visual feedback dominates the sense of agency for brain-machine actions. *PLoS ONE* **10**, e0130019 (2015).
37. Wodlinger, B. et al. Ten-dimensional anthropomorphic arm control in a human brain-machine interface: difficulties, solutions, and limitations. *J. Neural. Eng.* **12**, 016011 (2015).
38. Wang, F. et al. Enhancing clinical communication assessments using an audiovisual BCI for patients with disorders of consciousness. *J. Neural. Eng.* **14**, 046024 (2017).
39. Pfurtscheller, G., Guger, C., Müller, G., Krausz, G. & Neuper, C. Brain oscillations control hand orthosis in a tetraplegic. *Neurosci. Lett.* **292**, 211–214 (2000).

Acknowledgements

We thank the study participant and his family for their dedication and support. We thank D. Weber and H. Bresler for assistance in editing the manuscript; M. Zhang, S. Colachis, H. Trivedi, A. Singh, and P. Ganzer for their assistance with the sessions, data collection, and help editing the manuscript; and R. Kittel for help with figure formatting. Financial support for this study came from Battelle Memorial Institute and The Ohio State University Neurological Institute and Department of Physical Medicine & Rehabilitation. M.A.B. also acknowledges the invaluable mentorship of the Rehabilitation Medicine

Scientist Training Program at the Association of Academic Physiatrists. M.A.S. would also like to thank his brother E. Schwemmer for being a source of inspiration for this work.

Author contributions

D.A.F., G.S., M.A.S., and M.A.B. conceptualized the study; D.A.F., M.A.S., N.D.S., G.S., and M.A.B. designed the experiments; M.A.S., N.D.S., D.A.F., P.B.S., and J.E.T. performed research and data analysis; M.A.S., N.D.S., D.A.F., and M.A.B. wrote the manuscript; all authors contributed to editing the manuscript.

Competing interests

The authors declare competing interests, as they are employed by institutions that provided the funding for this work and/or have filed associated patents. M.A.S., N.D.S., J.E.T., D.A.F., and G.S. are all employed by Battelle Memorial Institute and M.A.B. is employed by the Ohio State University. P.B.S. was also employed by the Ohio State University at the time of this study. D.A.F. and G.S. are listed as inventors on the United States patent application US 2018/0178008 (related WO 2016/196797), and G.S. is listed as an inventor on the United States patent application US 2015/0306373. These are related to the neural bridging BCI technology and stimulation sleeve used in the GRT experiment in the paper.

Additional information

Supplementary information is available for this paper at https://doi.org/10.1038/s41591-018-0171-y.

Correspondence and requests for materials should be addressed to M.A.S.

Methods

Study design and participant details. This study (ClinicalTrials.gov: NCT01997125) was approved by the US Food and Drug Administration (Investigational Device Exemption) and the Ohio State University Wexner Medical Center Institutional Review Board (OSUWMC, Columbus, Ohio). The study conformed to institutional requirements for the conduct of human subjects. All experiments were performed in accordance with the relevant guidelines and regulations set by OSUWMC. The participant referenced in this work provided permission for photographs and videos and completed an informed consent process before commencement of the study.

The study participant is a man with C5 AIS A tetraplegia from traumatic SCI due to a diving accident that occurred 4 years before his enrollment in this study. He has no motor function below the level of C6. A Utah microelectrode array (Blackrock Microsystems, Inc., Salt Lake, Utah) was implanted in the hand area of his left primary motor cortex on 22 April 2014, as previously described[8].

In most of the experiments described in this manuscript, the participant viewed a computer monitor on which two animated hands were displayed (Fig. 1a). Cues to imagine specific movements were given by a small hand in the lower left corner of the screen. During rest periods, the cue hand remained in a neutral position. A larger hand centered on the screen provided real-time feedback from the BCI system during experiments in which the participant had volitional control and was given visual feedback. Otherwise, the feedback hand remained in a neutral resting position. The exceptions to this are when the participant was performing 'free-time' and GRT blocks described below. In these cases, the cue was provided by placing the object in front of the subject, and only the feedback hand driven by the participant using the decoder was displayed on the screen.

Datasets. The initial training dataset was based on a four-movement task in which the participant was cued to imagine either wrist flexion, wrist extension, index-finger flexion, or index-finger extension on each trial. Movement cues (2.5 s each) were shuffled and interleaved with rest periods (4 s each), as in Fig. 1b. One block consisted of four repetitions of each movement, for a total of 16 movements and 16 rest periods across 104 s. During each session, two consecutive blocks of data were collected with a brief rest period in between. In total, the participant performed this task 128 times from 18 July 18 2014 (60 d after he first used the BCI system) to 9 June 2017. Analyses in this paper were restricted to datasets collected between 26 January 2015 and 9 June 2017 in order to allow for participant recovery time from the surgery and adjustment to the overall BCI system set-up. This provided 80 sessions of data, representing 640 repetitions of each of the four movements across 16,640 s (~4.62 h). The first 40-session 'training period' spanned 484 d between 26 January 2015 and 24 May 2016. The subsequent 40-session 'testing period' spanned 373 d between 1 June 2016 and 9 June 2017 (Fig. 1c). The time from the last training day to the last testing day was 381 d.

The additional dataset used to train and test for transfer learning was based on a two-movement task in which the participant was cued to imagine opening or closing his hand on each trial. Movement cues (2.5 s each) were shuffled and interleaved with rest cues (6.5 s each). One block consisted of five repetitions of each movement, for a total of ten movement cues and ten rest periods across 90 s. Two blocks of data were collected at each session. In total, the participant performed this task 146 times from 19 May 2014 (the first day he used the BCI system) to 9 June 2017. Analyses in this paper were restricted to datasets collected between 1 June 2016 and 9 June 2017.

In both experiments, the participant received no feedback, limiting his ability to adapt to specific decoders and making these datasets ideal for testing different decoding algorithms.

Data acquisition and preprocessing. Data was collected at 30 kHz from the Utah microelectrode array with 1.5-mm electrodes using the Neuroport neural data acquisition (Blackrock Microsystems). A 0.3-Hz first-order high-pass and a 7.5 kHz third-order low-pass Butterworth analog hardware filters were applied to the data. The data were then transferred to a PC running Matlab R2014b and Python 2.7 for further processing.

The 30-kHz data were reduced using a wavelet decomposition with a 'db4' mother wavelet and 11 wavelet scales. Features for decoding were created by extracting the wavelet coefficients for scales 3, 4 and 5, spanning a frequency band from 234–1,875 Hz. Every 100 ms, the coefficients were averaged over time, providing $96 \times 3 = 288$ features for each 100-ms time bin. Next, these averaged wavelet coefficients for each channel were individually standardized by subtracting out the mean and dividing by the s.d. of each feature over a single block of data. During the training period, each block of data was standardized to itself, while during the testing period, the mean and s.d. of the first block was used to standardize both the first and second blocks. Once the 288 features were standardized, the three averaged and standardized coefficients for each channel were then averaged together to create a single feature, MWP, for each channel, resulting in 96 features for each 100-ms time bin. We used MWP features here as previous work from our group has demonstrated the success of their use in decoding movement intent[8,23,40,41]. Lastly, the MWP features for each channel were smoothed by replacing the current time point with the average of the most recent 10 time points (i.e., a 1-s boxcar filter). We have detailed the use of MWP in other work[8,23,40,41] and believe them to be excellent features for classification, but the NN models we use here could also be fit using other common features like threshold crossings, spikes, or local field potential (LFP).

To account for reaction and system lag times, the timing of the cues was shifted by 800 ms when training and calculating model accuracy. The cues were left unshifted when calculating the model failure rates and response times.

Decoding methods and training procedures. The training period data (40 sessions over the first year) were used to calibrate the fNN decoder, which acts as the starting model for all of the NN models. For each session during the testing period (40 sessions over the second year), the first block of data was used for updating two variants of the fNN, as well as calibrating a support vector machine (SVM) decoder unique to that day (Fig. 1d). The two variants of the fNN differed in how they were updated using the testing period data: either in a supervised manner, sNN, or in an unsupervised manner, uNN (see next section for more details on these different updating procedures).

The fNN was constructed and trained with Python 2.7 using the package Keras (https://github.com/keras-team/keras) with TensorFlow[42] as the backend. The model architecture is shown in Fig. 1e. It takes as input a 96×9-dimensional array corresponding to 900 ms of MWP data. That is, the model uses a sliding window of 900 ms to predict the imagined movement for the current time point. The first layer in the network is a long short-term memory (LSTM)[43] layer containing 80 hidden units. The LSTM is a variant of a recurrent NN which is widely used in the processing of temporal (or sequential) data[16]. The hidden units in the LSTM layer use the hard-sigmoid activation function whereas the output of the layer uses a hyperbolic tangent activation. The LSTM layer outputs an 80×9-dimensional array that is passed to a 1-dimensional convolutional layer that contains 25 filters that are 80×9-dimensional. The convolution is performed in the time domain only. The activation function for the output of the convolution layer is a leaky-rectified linear unit with an α parameter of 0.01. The output of this layer is then flattened to a 225-dimensional vector which is then passed to a fully connected (dense) layer with 50 units using the rectified linear unit activation function. The output from this layer is passed to a final dense layer containing five units corresponding to the four movements plus rest. The units in this final layer use the softmax activation function scaling the outputs to correspond to probabilities. The unit with highest probability is the predicted movement for that particular time-point, i.e., the argmax.

The fNN was trained using random batches of size 800 using the optimizer RMSprop[44] and the categorical cross-entropy loss function. All network parameters were randomly initialized at the start of the training using the Keras defaults. During each training epoch, each layer in the model underwent a random 50% dropout of the connections weights in order to prevent overfitting to the training data[45] (Fig. 1e). The training lasted for 100 epochs and took a total of 493.55 s to complete using an NVIDIA Quadro K5000 GPU on a Linux system running Centos 7.3. The two other NN models, sNN and uNN, both begin life as the fNN but differ in how they utilize data from the testing period for additional refinement.

The SVM classifier implementation closely follows that of refs. [8,23]. Specifically, the SVM uses nonlinear Gaussian radial basis functions kernels with a γ parameter value of 0.005, was trained using the MWP features extracted from the first block of data for a given session, and uses the MWP features at the current time point only in order to predict the intended movement. The SVM was trained using the sci-kit learn toolbox (http://scikit-learn.org/stable/) in Python 2.7 using the default parameter values, except for the value of the γ that we based on our previous experience using the SVM.

Supervised and unsupervised updating of the neural network model. The sNN involves updating the original fNN model in a supervised manner using explicit knowledge of the type and timing of movement cues during the first block of data from each session during the testing period. Supervised updating of the NN model occurred as follows: on the first session of the testing period, the MWP features and cues were extracted from the first block of data. This data was combined with the training period data to create a larger training dataset for the model. Next, the weights of the NN model were initialized to the fNN described in the previous section. This new model was then updated by running the same fitting procedure on the new training dataset for 30 epochs. The performance of this model was assessed using the second block of data. This process was repeated for each session during the testing period, with new data added to the updated training dataset every session, and the same fitting procedure applied to models initialized with weights from the previous session. The supervised aspect of this procedure refers to the fact that the actual cues used in the experiment were used in updating the model. This daily update was done to control for any effects of time and focus solely on optimizing the accuracy component of decoder performance.

The unsupervised updating procedure differed from the supervised procedure in the manner in which labels of cues were obtained. In the supervised case, the known experimental cues were used to label the MWP data from the first block. In the unsupervised case, the current model was first used to predict the cues for the first block of data for each session. That is, as data passes through the fNN, the predicted movement from the fNN model can be used to provide tentative labels for the data, creating additional training data that can be combined with the

historical data and used to update the model. The MWP data and the predicted cues were then combined with the training period data as described above. In addition, the fitting procedure was slightly different for the two types of updating: in the unsupervised case, the loss function was changed to the 'hard' boot strapping modification of the cross-entropy function first introduced by Reed et al.[46]. Briefly, this loss function allowed for the explicit inclusion of 'noisy' labels by modifying the regression target in the cross-entropy function to include a convex combination of the noisy label and the model's currently predicted label. Next, as in the previous paragraph, the fitting procedure with the new loss function was run for 30 epochs to update the model, and the process was repeated for each session during the testing period. Thus, the unsupervised aspect of this procedure refers to the fact that the labels for the data used in the updating were generated by the model itself and did not require the user to participate in a cued recalibration task.

This idea of unsupervised updating is related to the notion of semi-supervised self-training, which has been used to increase the accuracy of a naive Bayes object detection model[20], and has been used by Jarosiewicz et al.[17] for BCI using retrospectively supervised linear discriminant analysis (LDA) decoder calibration for point-and-click BCI systems. However, these models used the predicted labels as if they were the true label only if the prediction is made with sufficiently high confidence (to avoid contaminating the model with incorrectly labeled data). Additionally, a more recent updating method was proposed for cursor control in a BCI for virtual typing[3] that relies on retrospectively inferring the user's intended trajectory based on the character they selected on a virtual keyboard. In contrast, our unsupervised updating method leverages all of the tentatively labeled data directly in the updating procedure by weighting the influence of the tentatively labeled data proportional to the models' confidence in the predicted label[46] and is a general method that can be applied to any discrete classification decoding problem.

Performance measures. The primary metric we use to measure decoder performance is the percentage of 100-ms time bins in a test block in which the decoder correctly predicted the cued movement or rest. This metric is the standard accuracy metric for statistical classification algorithms. Note that the accuracy of the no-information decoder (the decoder that simply predicts rest since it is the most prevalent movement) is 61.54% on the four-movement task, as the rest cues appear in 640 time bins per block and the total number of bins in a block is 1,040. A locally weighted polynomial regression (LOESS) curve[47] with a smoothing parameter equal to 0.9 were added to accuracy plots to help better visualize the trend. We additionally introduce a failure rate metric, which was computed by tallying the percentage of times the model failed to predict the correct cue and maintain it for at least 1 s (10 time bins). The model response times were estimated by finding the first time point after cue onset where the correct movement was predicted and also sustained for at least 10 consecutive time bins (1 s). Note that for both the response time and failure rate, the first cued movement is ignored for both models because the uNN starts predicting during the middle of cue due to the time lagged input. From the failure rate metric, we also calculate a success rate by subtracting the failure rate from 100%. The failure and success rate metrics are meant to approximate an observer scoring each movement cue as a binary success or failure.

Transfer learning and fine-tuning of the NN model to increase the number of movements. Transfer learning and fine-tuning are methods of taking information from previously trained NN models and applying it to a new or refined task[21]. To create the dataset for this demonstration, experimental blocks collected on the same day from the four-movement and two-movement tasks were concatenated and treated as a new six-movement task (Supplementary Fig. 6).

Next, we initialized the transferred NN (tNN), a new NN model with analogous structure to the fNN except for the output layer, which had seven units (the previously used four movements, rest, and two new movements: hand open and hand close) instead of the previous five. The tNN was initialized using weights from the fNN for all layers except the output layer; weights for the tNN's output layer were initialized randomly. In the first stage of training, all the weights were held fixed, except the output layer weights that were calibrated using the categorical cross-entropy loss function with a batch size of 400 and were run for 30 epochs. This repurposing of the output layer of a NN model is referred to as transfer learning[21]. Once transfer learning was complete, we performed additional fine-tuning of the tNN: weights for the entire model were updated for an additional 20 epochs, resulting in a fully trained model for the combined six-movement task. Three separate versions of the utNN were trained by using both blocks of data of the combined six-movement task from either one, three, or five sessions (6.7 min, 20.2 min, or 33.7 min of total data, respectively).

During the transfer model testing period, the models were updated using the unsupervised procedure previously described, combining the six-movement training data with the predictions on the first block of the testing period (Supplementary Fig. 6). Each of the updated models was tested on the second block of data for the given session. For each subsequent session in the testing period, the predictions on the first block were added to the training set for the remainder of the testing period.

Real-time control of functional electrical stimulation experiment: grasp and release test. In these experiments, the decoder controlled FES of the participant's paralyzed forearm. The FES system consists of a multichannel stimulator and a flexible cuff consisting of up to 130 electrodes that is wrapped around the participant's arm. Offline, electrode stimulation patterns corresponding to the four movements in this task were calibrated using knowledge of forearm physiology and trial and error. The participant used the system to perform four movements (three functional grips and hand open) to manipulate three different objects (peg, fork, and can) from the standardized grasp and release test (GRT)[22], as shown in Fig. 4b. The decoder was used to predict the participant's intended movement every 100 ms, and the predictions were used to select the appropriate stimulation pattern, which was then actuated by the FES system to carry out the desired movement (Fig. 4a). The stimulation pulse rate was 50 Hz and the pulse width was 500 µs. Artifacts induced by the electrical stimulation were detected in the raw data by using threshold crossings of 500 µV occurring simultaneously on at least 4 of 12 randomly selected channels[8]. A 2.5-ms window of data around each detected artifact was then removed, and adjacent data segments were rejoined. For further details, please see refs [8,23,24].

In the first five blocks of the GRT experiment, the participant was cued to manipulate each of the three objects (can, fork, and peg). Prior to each cue, the object was placed in front of the participant. During the cue duration, the participant would grasp the cued object in the starting area and transfer it to an elevated platform using a combination of the system and his residual shoulder/elbow movement. In the case of the fork grip, the participant would grip the custom fork and apply downward pressure to cause cylindrical displacement. Additionally, during cued hand open, the participant would open his hand, which was necessary to position his hand to grip the can. Each movement cue had a random duration between 3 and 4 s and was bounded by rest cues with random durations between 5 and 6 s. The cues were presented in a random order, and each movement was repeated four times for a total of 128–160 s per block.

In the first block of the experiment, the subject received scripted stimulation corresponding to the cued movement. The data from this block was used to transfer and fine-tune the uNN model that had been updated up until 9 June 2017 in order to allow it to predict the new movements for this task (the three functional grips and hand open). That is, we initialized a new NN model with analogous structure to the uNN except for the output layer, whose units now corresponds to the three functional grips, hand extension, and rest. The new NN was initialized using weights from all layers of the uNN except the output layer; weights for the new NN's output layer were initialized randomly. In the first stage of training, all the weights were held fixed, except the output layer weights, which were calibrated using the categorical cross-entropy loss function with a batch size of 400 and run for 15 epochs. Once transfer learning was complete, we performed additional fine-tuning of the NN: weights for the entire model were updated for an additional 10 epochs. The transferred uNN model was then used to predict the cues and control the stimulation on the subsequent blocks.

The NN model was then used to control the FES system for the subsequent experimental blocks. For each block, the participant was instructed to grasp, transfer, and release the object as quickly as possible as soon as the cue began. After both of blocks 2 and 3, each model was updated in a supervised manner for 30 epochs. Thus, the model was calibrated using a total of three supervised blocks (147.5 s, 142.7 s, and 146.1 s; ~7.27 min total). Supervised updating was used, as this was a completely new task to which the model had not been previously exposed. Unsupervised updating was then performed after additional cued blocks 4 and 5 using the hard boot strapping modification of the cross-entropy function from Reed et al.[46] described in the Methods; this occurred for 15 epochs for each model. The output from block 5 is shown in Fig. 4d.

Next, the participant performed two 'free-time' blocks (blocks 6 and 7), in which he had to manipulate objects that were randomly placed in front of him. He was not cued on the computer screen during these blocks. The model was updated in an unsupervised manner following each of these blocks. The MWP data in the first block was standardized to itself, whereas the MWP data for all subsequent blocks were standardized using the data from the previous block.

The last three blocks of the experiment were the three trials of the GRT. Each block consisted of three runs (one for each object). A board-certified physiatrist administered the GRT in order to assess the functional grasps. For each object, the participant was required to grasp, transfer the object to an elevated platform, and release it as many times as possible in a 30-s test period. The participant was given 30 s of rest between each test period. Dropping the object (or insufficient cylinder displacement for the fork) was counted as a failure. For the fork, successful 'transfers' were counted if the spring-loaded piston was sufficiently displaced, indicated by a line on the piston. Two separate movements were required for the can transfer. The participant had to perform a hand open to position his hand in an optimal location around the can and then initiate the can grasp. The GRT was performed three times for each object. The NN model decoder and standardization were held fixed (not updated) across all three replicates. The participant's baseline performance on the GRT without using the BCI-FES system was collected on 24 March 2016 and 25 March 2016.

Comparison to existing FES solutions. As pointed out in Ajemian et al.[48], the invasiveness of collecting intracortical data necessitates justification that the invasive solution is better than noninvasive alternatives. The most comparable

system that does not require an invasive neurosurgery is the FreeHand system[49,50]. There are at least three relevant differences between the current system and the myoelectrically controlled FreeHand system. First, as members of the FreeHand team point out[9], the implanted microelectrode array BCI approach evokes FES-enabled movements more directly and intuitively than switches or myoelectric control. This decreases cognitive load and facilitates a more natural integration of the neuroprosthetic with end-user's body mechanics—he or she focuses directly on the grasp they want to make, rather than timing the surrogate movement (for example, shoulder shrug) with an evoked grasp. This is likely why members of the FreeHand team have recently been exploring BCI control of FES using multiple invasive Utah arrays[9].

A second difference between the FreeHand system and the one described here is the manner in which BCIs have been integrated with the FES component. Early integration of the FreeHand system under BCI, rather than myoelectric, control used either EEG or EcoG. The EEG solution (see ref. [51] for an example) linked movement-related brain potentials (event-related desynchronization or event-related synchronization) at Cz and C4 through an LDA classifier with phases of a single type of grip (lateral key). In this way, the authors demonstrated a limited ability to manipulate one of six objects (paperweight) in the GRT task. In our work, in large part because of the richer signal from the implanted array, we have demonstrated simultaneous control of a large number of varied movements[8,23,26,40,52] as well as continuous graded control of muscle contractions[24].

The third difference between our approach and the FreeHand approach is type of FES used. The FreeHand system is implanted in muscle, whereas ours is transcutaneous and applied via wearable sleeve. These are different FES solutions that differentially address competing expectations and needs of end-users. We believe the decoding paradigm we propose here could be adapted to control the FreeHand FES system.

Statistical tests. Two-tailed paired Wilcoxon signed rank tests were performed for all between-model comparisons of accuracy and response times. An unpaired two-tailed Wilcoxon signed rank test was used to compare the transfer times for peg in the GRT experiment. All tests were performed in the R programming language.

Reporting Summary. Further information on research design is available in the Nature Research Reporting Summary.

Code availability. Code used in this study can be made available to qualified individuals for collaboration provided that a written agreement is executed in advance between Battelle Memorial Institute and the requester's affiliated institution. Such inquiries or requests should be directed to D.A.F.

Data availability

Data used in this study can be made available to qualified individuals for collaboration provided that a written agreement is executed in advance between Battelle Memorial Institute and the requester's affiliated institution. Such inquiries or requests should be directed to G.S.

References

40. Friedenberg, D. A. et al. Big data challenges in decoding cortical activity in a human with quadriplegia to inform a brain computer interface. in *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* 3084–3087 (IEEE, Piscataway, NJ, USA, 2016).
41. Sharma, G. et al. Time stability and coherence analysis of multiunit, single-unit and local field potential neuronal signals in chronically implanted brain electrodes. *Bioelectron. Med.* **2**, 63–71 (2015).
42. Abadi, M. et al. TensorFlow: large-scale machine learning on heterogeneous distributed systems. Preprint at https://arxiv.org/abs/1605.08695 (2016).
43. Hochreiter, S. & Schmidhuber, J. Long short-term memory. *Neural Comput.* **9**, 1735–1780 (1997).
44. Ruder, S. An overview of gradient descent optimization algorithms. Preprint at https://arxiv.org/abs/1609.04747 (2016).
45. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**, 1929–1958 (2014).
46. Reed, S. et al. Training deep neural networks on noisy labels with bootstrapping. Preprint at https://arxiv.org/abs/1412.6596 (2014).
47. Cleveland, W. S. & Devlin, S. J. Locally weighted regression: an approach to regression analysis by local fitting. *J. Am. Stat. Assoc.* **83**, 596–610 (1988).
48. Ajemian, R. Neurosurgery: gentler alternatives to chips in the brain. *Nature* **544**, 416 (2017).
49. Peckham, P. H., Mortimer, J. T. & Marsolais, E. B. Controlled prehension and release in the C5 quadriplegic elicited by functional electrical stimulation of the paralyzed forearm musculature. *Ann. Biomed. Eng.* **8**, 369–388 (1980).
50. Taylor, P., Esnouf, J. & Hobby, J. The functional impact of the Freehand System on tetraplegic hand function. Clinical Results. *Spinal Cord* **40**, 560–566 (2002).
51. Müller-Putz, G. R., Scherer, R., Pfurtscheller, G. & Rupp, R. EEG-based neuroprosthesis control: a step towards clinical practice. *Neurosci. Lett.* **382**, 169–174 (2005).
52. Friedenberg, D. A. & Schwemmer, M. A. Moving a paralyzed hand—a biomedical big data success story. *Chance* **29**, 4–13 (2016).

nature research

Corresponding author(s): Michael Schwemmer

Life Sciences Reporting Summary

Nature Research wishes to improve the reproducibility of the work that we publish. This form is intended for publication with all accepted life science papers and provides structure for consistency and transparency in reporting. Every life science submission will use this form; some list items might not apply to an individual manuscript, but all fields must be completed for clarity.

For further information on the points included in this form, see Reporting Life Sciences Research. For further information on Nature Research policies, including our data availability policy, see Authors & Referees and the Editorial Policy Checklist.

Please do not complete any field with "not applicable" or n/a. Refer to the help text for what text to use if an item is not relevant to your study.
For final submission: please carefully check your responses for accuracy; you will not be able to make changes later.

Experimental design

1. Sample size

   Describe how sample size was determined.

   We have a single participant enrolled in our clinical study who has undergone the surgical procedure to have a microelectrode array implanted in his motor cortex.

2. Data exclusions

   Describe any data exclusions.

   No data were excluded in the analyses presented in this paper. The analyses included all experimental datasets collected between January 26, 2015 and June 9, 2017. Data collection details are described in depth in the Methods section of the manuscript.

3. Replication

   Describe the measures taken to verify the reproducibility of the experimental findings.

   Data is collected from the participant in a standardized format every session. Additionally, the parameters of the trained fixed neural network and support vector machine models were saved so that their outputs can be exactly reproduced. The output of the neural network models that use supervised or unsupervised updating might vary slightly owing to stochasticity in the updating procedures (i.e., we do not save the updated models for each individual session). However, given the parameters of the fixed neural network as the starting point, the outputs of the updated models will be similar to the results we show in the manuscript.

4. Randomization

   Describe how samples/organisms/participants were allocated into experimental groups.

   A single participant is enrolled in this clinical study, thus preventing the ability to randomize.

5. Blinding

   Describe whether the investigators were blinded to group allocation during data collection and/or analysis.

   Blinding was not used in this study due to the single participant and single treatment design. Given the invasive nature of the system, the participant is usually well aware of small changes to the system, preventing us from making changes without his knowledge. In all of the experiments demonstrating the performance of the neural network, the validity of the results are not biased by the participants knowledge of the decoder type.

Note: all in vivo studies must report how sample size was determined and whether blinding and randomization were used.

6. Statistical parameters

For all figures and tables that use statistical methods, confirm that the following items are present in relevant figure legends (or in the Methods section if additional space is needed).

n/a	Confirmed
☐	☒ The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement (animals, litters, cultures, etc.)
☐	☒ A description of how samples were collected, noting whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☐	☒ A statement indicating how many times each experiment was replicated
☐	☒ The statistical test(s) used and whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☐	☒ A description of any assumptions or corrections, such as an adjustment for multiple comparisons
☐	☒ Test values indicating whether an effect is present *Provide confidence intervals or give results of significance tests (e.g. P values) as exact values whenever appropriate and with effect sizes noted.*
☐	☒ A clear description of statistics including central tendency (e.g. median, mean) and variation (e.g. standard deviation, interquartile range)
☐	☒ Clearly defined error bars in all relevant figure captions (with explicit mention of central tendency and variation)

See the web collection on statistics for biologists for further resources and guidance.

Software

Policy information about availability of computer code

7. Software

Describe the software used to analyze the data in this study.

A PC running MatlabR2014b was used for data collection and custom scripts were used to convert the raw voltage recordings from the microelectrode array into the mean wavelet power features described in out Methods section. Analysis and model fitting were done on a PC using Python 2.7. All neural network models were trained and tested within python using the Keras package (v1.2.2) with Tensorflow (v1.0.1) as the backend. The support vector machine classifier was also constructed in python using the sci-kit learn package (v0.17). Python scripts were created to load the data, construct and train the classifier models (using Keras), and test the output. Scripts in R using the ggplot package were used to plot the results used in the figures.

For manuscripts utilizing custom algorithms or software that are central to the paper but not yet described in the published literature, software must be made available to editors and reviewers upon request. We strongly encourage code deposition in a community repository (e.g. GitHub). *Nature Methods* guidance for providing algorithms and software for publication provides further information on this topic.

Materials and reagents

Policy information about availability of materials

8. Materials availability

Indicate whether there are restrictions on availability of unique materials or if these materials are only available for distribution by a third party.

Materials used in this study (data and/or code) can be made available to qualified individuals for collaboration provided that a written agreement is executed in advance between Battelle Memorial Institute and the requester's affiliated institution. Such inquiries or requests should be directed to G.S.

9. Antibodies

Describe the antibodies used and how they were validated for use in the system under study (i.e. assay and species).

No antibodies were used in this study.

10. Eukaryotic cell lines

a. State the source of each eukaryotic cell line used.

No eukaryotic cell lines were used.

b. Describe the method of cell line authentication used.

No eukaryotic cell lines were used.

c. Report whether the cell lines were tested for mycoplasma contamination.

No eukaryotic cell lines were used.

d. If any of the cell lines used are listed in the database of commonly misidentified cell lines maintained by ICLAC, provide a scientific rationale for their use.

No eukaryotic cell lines were used.

Animals and human research participants

Policy information about studies involving animals; when reporting animal research, follow the ARRIVE guidelines

11. Description of research animals

Provide all relevant details on animals and/or animal-derived materials used in the study.

No animals were used in this study.

Policy information about studies involving human research participants

12. Description of human research participants

Describe the covariate-relevant population characteristics of the human research participants.

The participant is a 27-year-old male with stable, non-spastic tetraplegia from a cervical spinal cord injury that he suffered at the age of 19. His injury was complete, with an overall neurologic level of C5 AIS A with zone of partial preservation to C6. A Utah microelectrode array (Blackrock Microsystems, Inc., Salt Lake, Utah) was implanted in the hand area of his left primary motor cortex on April 22, 2014.